HERA #134
Recieved 28 May 2026

HERA Team Memo: H6C Internal Data Release 3.1

Joshua S. Dillon, Tyler A. Cox, Steven G. Murray for the HERA Analysis Team

May 28, 2026

1 Summary

In this memo, we lay out the key updates to HERA analysis since H6C IDR2.3 (Dillon et al., 2024), the analysis on which our first Phase II limits (HERA Collaboration et al., 2025) were based. These include substantial new analyses for finding RFI after a corner-turn to single-baseline files and improved inpainting using a scaffold with feathered weights (see subsection 3.3). There is also a new LST-stacker designed to operate on single-baseline files while using the same new inpainting techniques to find and correct outliers (see section 4). We also provide a summary of data products to help orient consumers of this release (see section 6) and enumerate open questions and future directions (see section 7).

2 Data Selection

This analysis began with the 147 nights identified in Murray and Dillon (2023) out of the 177 total nights with data in H6C. The first night is 2459847 (September 24, 2022) and the last night is 2460038 (April 3, 2023). During analysis, the following nights were cut:

- 2459851: too much broadband RFI in the middle of the night.
- 2459886: too much broadband RFI in the beginning and middle of the night.
- 2459892: wide RFI spike around 137 MHz (ORBCOMM), plus much of the night was previously flagged after initial analysis.
- 2459896: unusually high temporal instability in the gains in the nn polarization.
- 2459897: unusually high temporal instability in the gains in the nn polarization.
- 2459913: too much broadband RFI in the beginning and end of the night.
- 2459915: too much RFI in the beginning and middle of the night, plus in the TV band.
- 2459919: unusually high temporal instability in the gains in both polarizations.
- 2459924: too much broadband RFI throughout the night.
- 2459928: too much broadband RFI in the beginning and middle of the night.
- 2459959: instability in gains in both pols caused too many antennas to be flagged.
- 2459980: too much broadband RFI throughout the night.
- 2460021: too much broadband RFI at the beginning and middle of the night.
- 2460023: every other file is missing or was never taken.
- 2460037: too much RFI in the beginning of the night, plus in the TV band.

This left a total of 131 nights to integrate up.

3 Per-Night Analysis

Per-night analysis is responsible for the flagging of bad antennas and RFI, per-antenna gain calibration and smoothing, redundant-averaging, and a corner-turn to single-unique-baseline, 4-pol, full-night files (as opposed to the raw data, which is made of two-integration, all-baseline files). Most of the updates to the per-night pipeline compared to IDR 2.3 (Dillon et al., 2024) come after that corner turn, which we discuss in subsection 3.3. In Figure 1 and Figure 2 we show the entirety of the per-night pipeline and data flow, the various notebooks that constitute it, and how data flows through it. The pipeline shown in Figure 1 is specified in `h6c_analysis.toml`. The pipeline shown in Figure 2 is specified in `post_inpaint_analysis.toml`.

3.1 Updates to Redundant Calibration

While calibration smoothing (see subsection 3.2) is designed to handle occasional blips in calibration solutions, either due to `redcal` or `abscal`, enough errors can overwhelm the smoother and produce unreliable results. Partway through this analysis, we found this to be the case on a significant number of days, including 2459906, 2459907, 2459908, 2459909, 2459910, 2459912, and 2459976.

Digging into the problem, we discovered a situation where our previous technique for iteratively calibrating and throwing out high χ^2 antennas got the solution stuck in a bad place, which then caused `abscal` to find a bad solution for the whole polarization.

Previously, we would first perform `firstcal` and `redcal` for all antennas for 50 iterations, then flag all antennas with *median* $\chi^2 > 15$ (over frequency and both integrations). By design, the degeneracies are set by `firstcal` until we try to fix them with `abscal`. Next, we'd run several rounds of `redcal` with 50 iterations each, *starting from the previous set of solutions*. In each round, we'd throw out all antennas with *mean* $\chi^2 > 3$ (ignoring RFI-flagged channels)—unless that antenna's χ^2 was more than 1.5 times less than the worst antenna. This was designed to prevent good antennas getting thrown out due to bad neighbors artificially raising their χ^2. Unfortunately, this approach had three interrelated problems: first, several antennas could get thrown out in the first step that actually shouldn't have been because there was no dynamic range. Second, by keeping the solution from round to round, we could get stuck in a local minimum of the optimization landscape unfortunately driven by bad antennas. Finally, the combination of this bad solution with a bad set of `firstcal` degeneracies led us to get wildly different `abscal` solutions from integration to integration, creating additional temporal structure.

So, for those days and all days afterwards, we took a different approach. We got rid of that $\chi^2 > 15$ step entirely. Instead, we do rounds of $\chi^2 > 3$ and dynamic range cuts on antennas based on the *median*. Then we do the same thing based on the *mean* (though this only rarely throws out antennas). This two-step process is a bit more expensive than the one we invented for H6C IDR 2.2 (Dillon and Murray, 2023b), but it is more reliable. Then we throw away the solution entirely, starting `firstcal` again, and run the calibration for 250 iterations[1] without additional antenna flagging. This seems to produce much more stable solutions.

There is still more temporal variability in the solutions coming out of the `file_calibration` notebook than is ideal. And frankly, this solution is a bit uncomfortably *ad hoc*. Likely, this is something we may want to address in the future, perhaps with a data-driven sky calibration (see section 7).

3.2 Updates to Calibration Smoothing

We've made three important updates to calibration smoothing. The first is to the logic in the discovery of phase flips. Some antennas perform spontaneous 180° phase flips from one integration to the next, likely due to a malfunctioning Walsh switcher. Originally, these were detected by looking for large deviations from the antenna's starting phase, but we discovered that gain instability due to a poor `abscal` model (particularly around 8 hrs LST near the Galactic anti-center) can cause phase drift that could trigger that threshold. So we moved instead to looking for $> 90°$ phase shifts from integration to integration.

Unfortunately, the instability we described in subsection 3.1 can trigger this new criteria. Worse yet, if that instability causes a, let's say, 120° shift (above threshold) followed by two −60° shifts (both below threshold), we could end up in a state where the algorithm surmises that a permanent phase shift has

[1] Due to a parameter passing error, a few days after those indicated only had 50 iterations at this point. Most were 250.

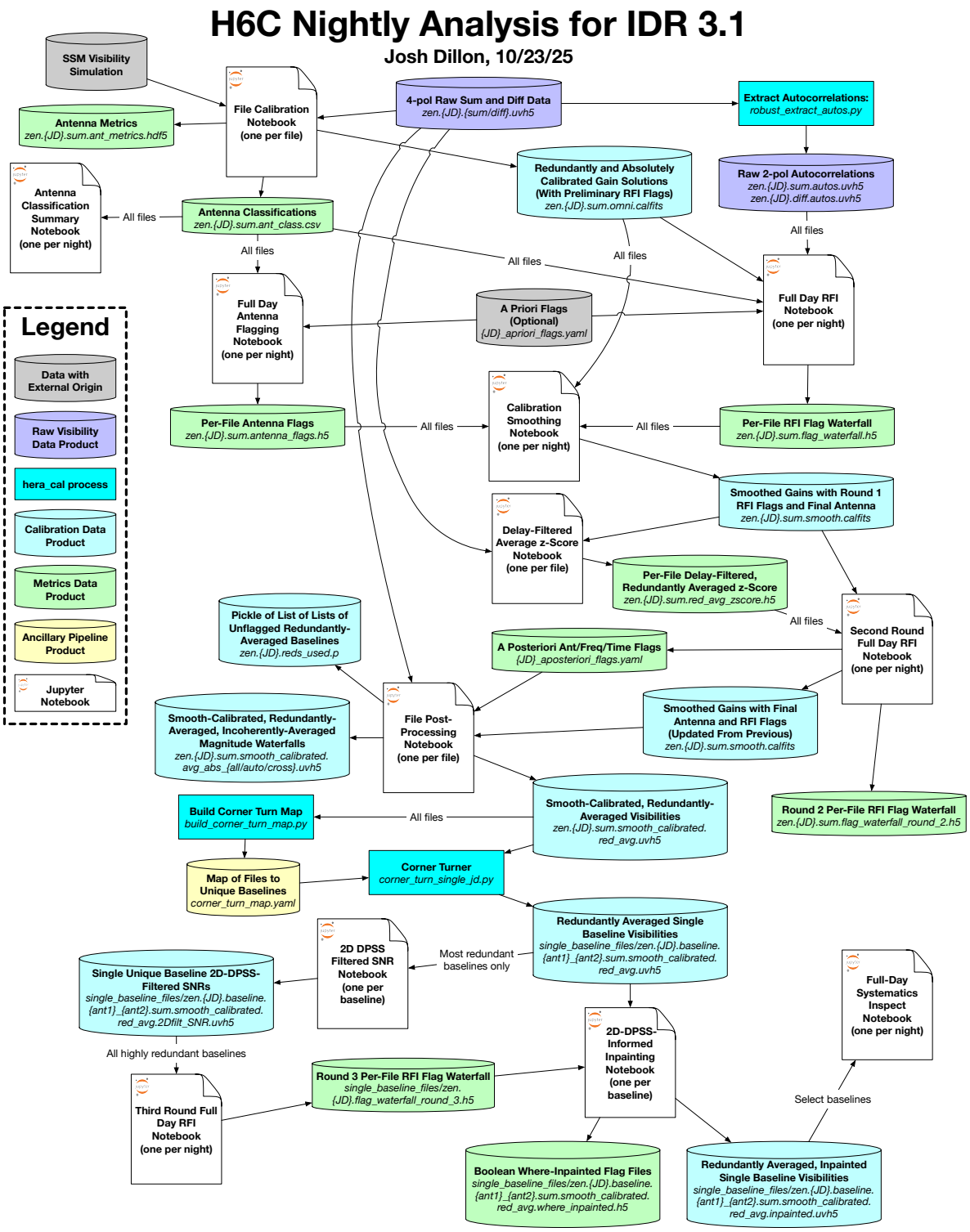

Figure 1: The first part of the updated pipeline for H6C IDR 3.1. Just like in IDRs 2.1, 2.2, and 2.3 (Dillon and Murray, 2023a,b; Dillon et al., 2024), the per-day analysis is centered on a series of notebooks, either one per file or just one per night, each of which both performs an analysis task and visualizes representative results in plots and tables. All the per-night notebooks, as well as the per-file notebook corresponding to the middle file of each night, are available at https://data.nrao.edu/hera/Notebooks/H6C_IDR3/. The a priori flags come from the "scouting" discussed in Murray and Dillon (2023).

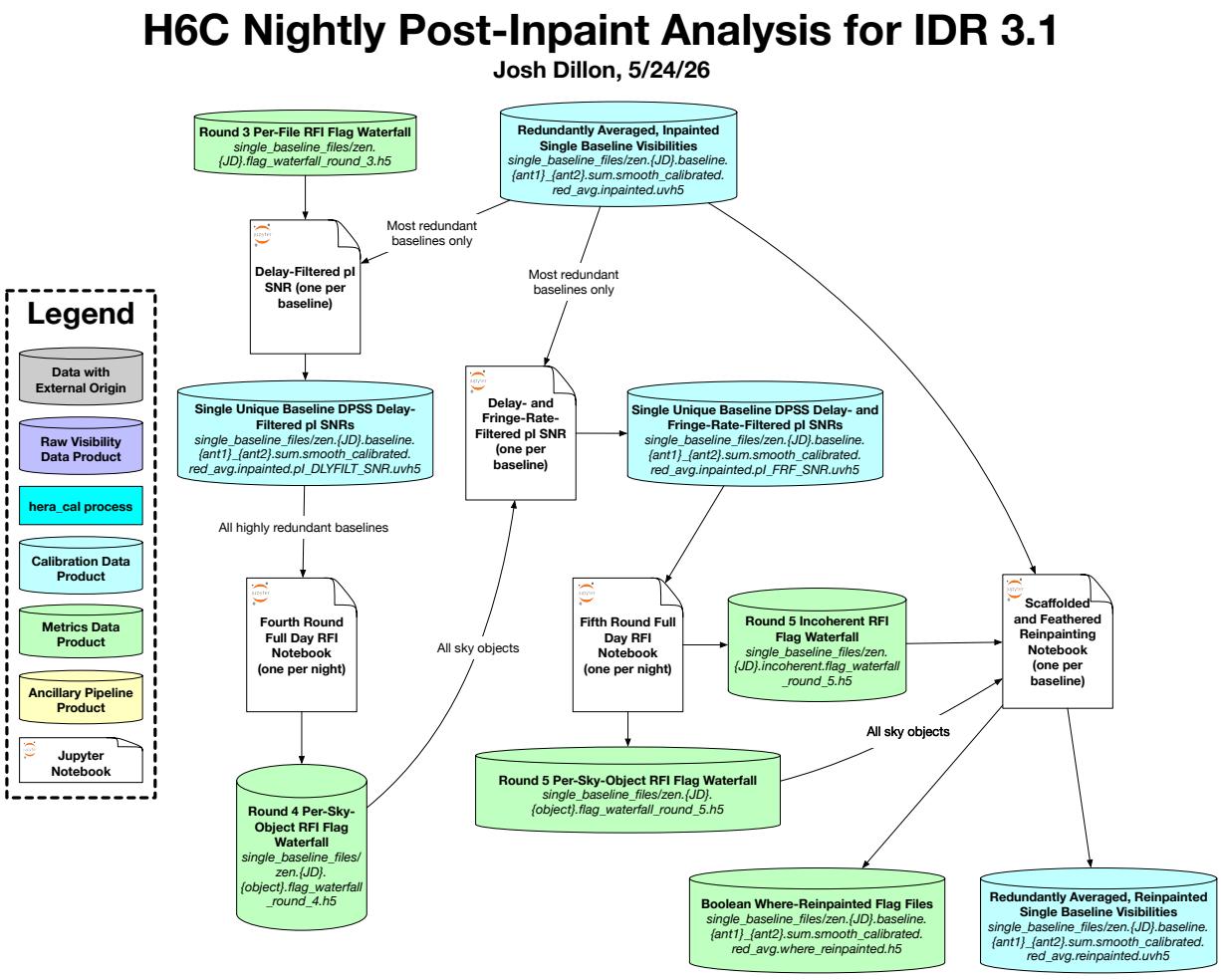

Figure 2: The second part of the per-night pipeline updated for H6C IDR 3.1. This was developed subsequent to running the first part shown in Figure 1, which is why we decided to add these steps onto the end, even though it is not optimal to inpaint, flag, and then re-inpaint. Future pipelines will better integrate these steps. All the per-night notebooks, as well as the per-file notebook corresponding to the middle file of each night, are available at https://data.nrao.edu/hera/Notebooks/H6C_IDR3/.

occurred, introducing a significant temporal discontinuity that cannot be smoothed over (recall that we undo the flip before smoothing and then re-apply it afterwards).

To solve both of these problems, we devised a third goldilocks algorithm that has the best features of the previous two. As described in this PR, we took out slowly varying phase drift with a 1D DPSS fit not to $g_i/|g_i|$ but to $g_i^2/|g_i|^2$. This quantity is agnostic to phase flips, since a 180° phase flip in g_i would be a 360° phase change in the gain squared. Of course, once we have that fit, we have to take the square root and decide whether the solution or the negative of the solution is a better fit, but this is trivial. The result is then a reliable way to remove the slow drift, which makes the identification of real phase flips easier and more reliable.

The other two changes are also related to addressing temporal (and potentially spectral) instability in gains described in subsection 3.1, which were potentially exacerbated by some of the changes to smoothing scales (we made the time scale shorter and the frequency scale longer, see subsection 3.4). In order to not overfit bad times, we first identify particularly "unsmooth" patches in gains by computing the average relative difference over antennas $\langle (g_i - g_i^{\rm smooth})/|g_i^{\rm smooth}| \rangle_i$. This is done on a per-polarization basis and this initial smoothing is done with a time-scale 4 times longer than the one we intend to use for smoothing. Any times and frequencies with average discrepancies greater than 100% are "blacklisted" and given 10%

weighting in the final 2D DPSS fit. This helps identify consistent patches of bad gains, usually from `abscal` problems, and prevent them from dragging the final calibration solutions around too much. If after all of this the smoothed gain is still very different from the unsmoothed gain (i.e. if $\langle (g_i - g_i^{\rm smooth})/|g_i^{\rm smooth}| \rangle_{t,\nu} > 20\%$), we simply throw out the antenna.

Finally, we tweaked the timescales to better match what we judged to be intrinsic variability in the gains. This led to smoothing on larger frequency scales (30 MHz rather than 10 MHz, though with a very small eigenvalue cutoff in DPSS this isn't actually that many fewer modes). Likewise, we smoothed in smaller timescales (10^4 s rather than 6×10^5 s). This seemed to do a better job of avoiding instances of really low gain amplitudes after smoothing, which can lead to very large visibilities. Exploring this parameter choice (and the role of calibration smoothing more broadly) remains a key question for future analyses.

3.3 New post-corner-turn analysis

The remainder of the per-day analysis pipeline takes place after redundant baseline averaging and "corner-turning," which in this context means going from files that contain all unique baselines and two integrations to files that contain a single unique baseline (with all four polarizations) and all times on a night. (See Figure 1 for how this step fits in the broader context of per-night analysis.) This process is I/O-limited and somewhat slow, but once we bite this bullet, we never have to corner-turn back. This required us to redesign LST-Cal (see subsection 4.1) and LST-stacking (see subsection 4.2), though it actually simplified some aspects of both of those. More importantly, it allowed for processes that better use temporal structure in the data both to find RFI (see subsubsection 3.3.1) and to inpaint missing data (see subsubsection 3.3.3).

3.3.1 Round 3 RFI Flagging: Flagging High-Delay, Non-Skylike RFI

Round 3 RFI flagging borrows many of the ideas from Round 2 RFI flagging (Dillon and Cox, 2024; Dillon and Murray, 2023b). Fundamentally, we are performing a high-pass DPSS filter of the highest-redundancy visibilities, normalizing by the noise to get a z-score, and then combining that quality incoherently across baselines to get a notion of outlier-ness for each time and frequency. This is used to produce a single baseline- and polarization-independent flagging waterfall. The key difference is that here the high-pass filter is performed with 2D-DPSS modes.

Like in Round 2, we remove power below 750 ns in delay. We also remove power outside the set of fringe-rates consistent with the sky at any frequency inside the band of interest calculated with `hera_cal.frf.sky_frates()` (below FM and above FM are filtered separately). Because temporal evolution of the visibilities comes not only from sky rotation but also from the passage of sources through the primary beam, we fit the simulated fringe-rate spectrum of the autocorrelations (which is dominated by sources moving through the primary beam) with a Gaussian and then buffer by a width equivalent to conservatively taking that Gaussian out to a height of 10^{-5}. The simulated spectrum is the same one used for Pascua et al. (2024) and in calculating main-beam fringe-rate filters of LST-stacked data prior to power spectrum estimation.

Like in Round 2, we perform a correction for leverage, without which points near flags would have erroneously low z-scores because the DPSS fits there are simply less constrained by the data. In Round 2, we calculated this effect exactly, as laid out in Dillon and Cox, 2024. In Round 3, the exact calculation is much more computationally difficult, but an approximate calculation (where flags are treated as separable in time and frequency) is still feasible. That calculation is done in the `single_baseline_2D_filtered_SNRs` notebook, which includes a re-scaling to approximately account for additional flags that are not included in entirely flagged integrations or entirely flagged channels.

After the 2D DPSS filter was performed, we found it useful to perform a subsequent very-low-delay 1D DPSS filter, which helps prevent flagging whole integrations due to temporal structure (often related to antenna flagging, which can turn on and off on a 2-integration timescale). This was performed at 200 ns below FM and at 50 ns above FM (though, given the differential delay resolution, a similar number of modes are filtered). Before 1D filtering, 8 MHz chunks on single baselines in TV bands starting at 174 MHz with average magnitudes of per-channel z-scores greater than 1 are given 0 weight, which usually results in them being even stronger outliers after 1D DPSS filtering. The identification of RFI from TV was rather poor in Round 2 flagging—the 750 ns filter removed most of the power. Here, that flagging is much more reliable,

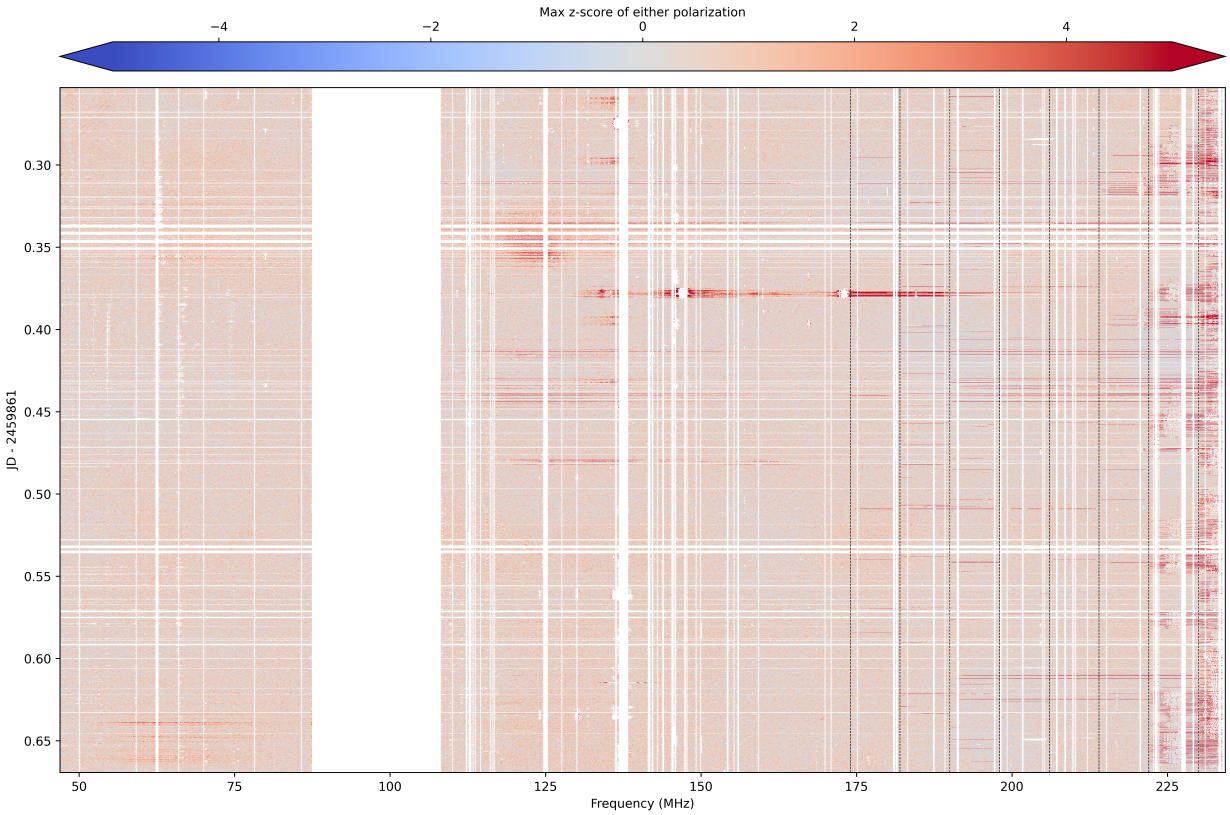

Figure 3: Even after Round 2 flagging, additional relatively broadband RFI leaks through, as we see here for 2459861. This is caught when we move to using 2D DPSS filtering, which allows us to find structure that was previously largely absorbed by the 750 ns 1D DPSS filter, but is clearly non-sky-like in its temporal structure. It is not completely clear whether every high z-score point here is RFI. Some appear clearly associated with known RFI. Some may be filtering artifacts. Some may be mutual coupling. Further investigation is warranted.

since the TV RFI seems to have considerable temporal structure (perhaps due to over-the-horizon bounces off airplanes or ionized meteor trails).

After all the single-baseline z-scores are brought together, a final decision about the overall flagging waterfall is made in the `full_day_rfi_round_3` notebook. This is similar to Round 2 flagging—in both instances, we bring together many baselines to create a new z-score which is more sensitive than any single baseline. An example from 2459861 is shown in Figure 3.

However, there are a few differences which ought to be enumerated. First, a decision is made about whether to flag entire 8 MHz TV allocations in a given integration. This is done by flagging allocations with average z-scores > 1 (excluding those already flagged and those with $|z\text{-score}| > 10$). The second difference is that after the first round of watershed flagging, there is an iterative frequency-convolved flagging step. This looks for N-channel stretches whose average z-score exceeds some threshold that gets lower as the stretch gets larger. Based on the current parameter choices, this currently goes from the 1-channel threshold of 5σ to 3 channels (4.757σ) to 5 channels (4.525σ) to 9 channels (4.096σ) to 17 channels (3.355σ) to 33 channels (2.251σ) to 65 channels (1.013σ). This is followed by another watershed flagging step before proceeding to mean-based whole-integration or channel flagging, as in Round 2.

The result of all this flagging is on the z-scores in Figure 3 is shown in Figure 4. While some of the choices are a little *ad hoc*, fundamentally the process was tweaked to help pick out coherent structures in z-score that were clearly visible by eye but difficult to pick out with simpler threshold cuts. It should be noted that this is a substantial increase in flagging and creates some rather wide flagging gaps that are very difficult to inpaint with the techniques we have used up to this point. If we were to keep the inpainting

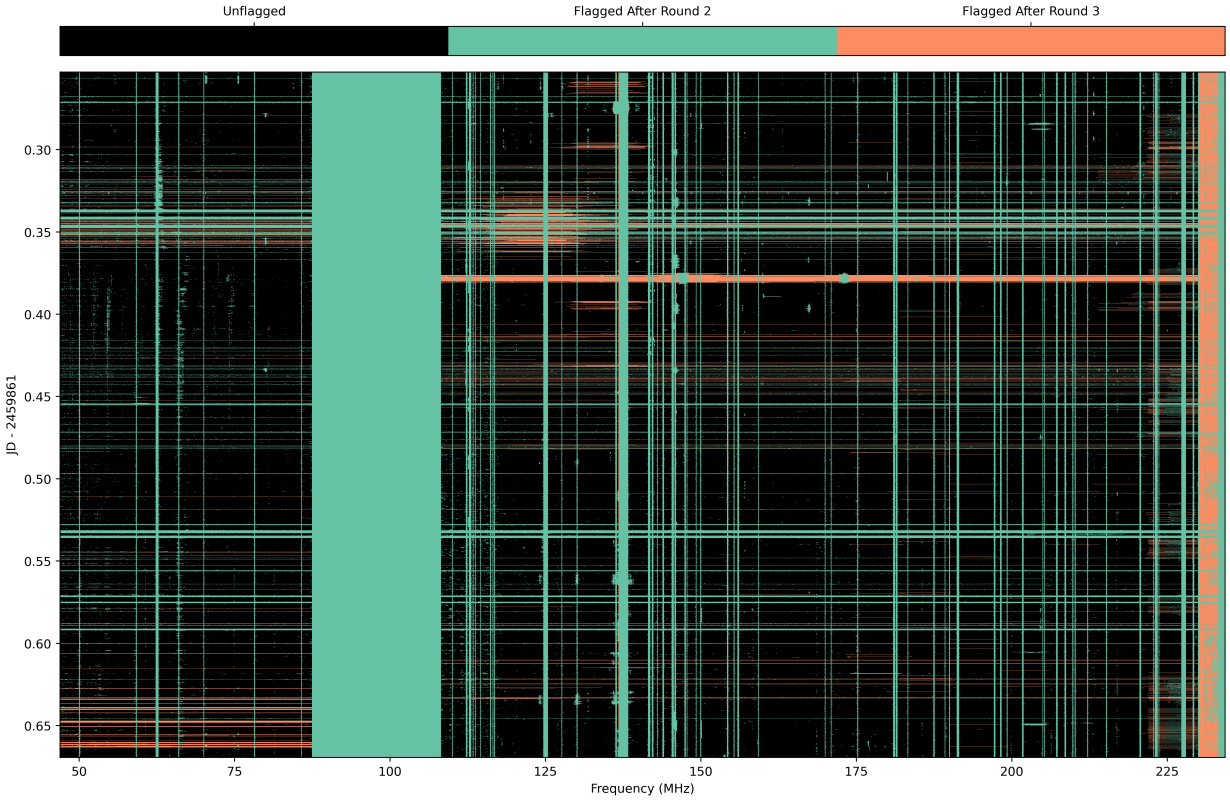

Figure 4: As a result of the high z-score times and frequencies found by 2D DPSS filtering (see Figure 3), additional Round 3 RFI flags are warranted, as we show here for 2459861. We employ largely similar algorithms for flagging on z-score as we did in Round 2, albeit with some differences tailored to the kind of RFI we are now detecting. See subsubsection 3.3.1 for details. The addition of large flagging gaps further motivates an inpainting algorithm more robust to pop-ups, which we discuss in subsubsection 3.3.2 and subsubsection 3.3.3.

and flagging strategy for HERA Collaboration et al. (2025), we would end up with an even higher level of flagging and need to give up on many power spectrum bands. A better approach is needed, which we address in subsubsection 3.3.2 and subsubsection 3.3.3 below.

Before we move on, we should note that Round 3 flagging has revealed a wide array of RFI phenomena that we do not understand (and are thus conservatively flagging). For example, most nights get pretty strongly flagged above 230 MHz. It is not clear if this is TV, instrumental, algorithmic, or something else. This is no great loss: we do not really expect to be able to do cosmology at that frequency anyway. But there are many other sorts of phenomena flagged as RFI that may or may not be RFI, including:

- Lots of extra emission coming off ORBCOMM on e.g., 2459880.
- Single huge bursts apparently coming from ORBCOMM on e.g., 2459946.
- Fairly narrowband blips near ORBCOMM, but not obviously associated with it, on e.g., 2459890.
- Lots of additional broadband RFI, likely due to lightning, on e.g., 2459886.
- A weird "X" feature around 125 MHz in e.g., 2459912 or 2459898. Is it mutual coupling? Aliasing? Something else?
- Parallel diagonal lines below FM in e.g., 2459847 or 2459853.
- Features that look like dispersed bursts below FM in e.g., 2459950, though there are also similar features that look like bursts except with negative DM in e.g., 2459870 and 2459894.

- Short duration spikes above 150 MHz that last only ∼1 hour on e.g., 2459953. These are mostly caught by Round 2 flagging, but perhaps not entirely.

This is merely a rogue's gallery. A full set of Round 3 outputs can be found here.

3.3.2 Inpainting with Feathering

One major challenge of inpainting is that the more degrees of freedom one allows a, e.g. DPSS fit, the more likely the inpainted visibility is to have a substantial deviation from the truth when flag gaps are wide—i.e. a "pop-up." These can then introduce unwanted spectral structure out to the inpaint delay limit, as well as night-to-night variability. Our previous attempt to resolve this problem, which we called "simultaneous inpainting" tried to minimize a DPSS model's deviation from both the data on the night in question and from data on other nights (Dillon et al., 2024).

However, subsequent to Dillon et al., 2024, we discovered that this technique introduced low-level spectral structure that became particularly apparent in the most-inpainted subbands. Basically, the penalty for deviating from other nights at the boundary between real and inpainted data caused discontinuities, especially in outlier nights. Even with LST-cal (see subsection 4.1), this problem was a show-stopper. Thus, for our first Phase II limits (HERA Collaboration et al., 2025), we backed off from the aggressive inpainting strategy, deciding that the existence of a sufficiently large flagging gap in a particular integration in a particular power spectral window would simply cause us to flag the entire window for the entire integration at the point of LST-stacking. This caused lost sensitivity, made it difficult to inpaint to higher delay (which shortens the maximum permissible gap), and essentially locked in our choice of spectral windows.

For this IDR, we have developed a new technique we call "informed inpainting with feathering" which enables substantially more aggressive inpainting (over large gaps and out to 1000 ns) while minimizing pop-ups. This technique is used both for the per-night inpainting of single unique baseline files post-corner-turn, as well as re-inpainting during LST-stacking (see subsection 4.2 below). The basic idea is to give the highest weight to real data, so as to ensure spectral smoothness of inpainted visibilities, but to give increasing weight to some external guess of the data—a "scaffold" data set—to inform the inpaint as we get further and further from the real data.

That is to say, we construct a delay-limited DPSS model fit to the data at a given time (or LST), where we have it, and to some other scaffolding visibilities where we don't:

$$V_{ij}^{\text{to fit}}(\nu) = \begin{cases} V_{ij}^{\text{data}}(\nu) & \text{if } \xi_{ij}(\nu) = 1 \\ V_{ij}^{\text{scaffold}}(\nu) & \text{if } \xi_{ij}(\nu) = 0 \end{cases} \tag{1}$$

where $\xi_{ij}(\nu)$ is an indicator function of whether or not the real data is flagged. Inpainting is then performing a least-squares fit to $V_{ij}^{\text{to fit}}(\nu)$ with some parameterized model and then using the results of that fit to substitute in for flagged data and to "unflag" it.

In traditional inpainting, what one puts into the flagged channels doesn't matter because they receive zero weight in the DPSS fit:

$$W_{ij}(\nu) \propto \frac{N_{ij}(\nu)\xi_{ij}(\nu)}{V_{ii}^2} \tag{2}$$

where V_{ii} is the autocorrelation (likely also inpainted, albeit to a lower delay because autocorrelations are so smooth) and $N_{ij}(\nu)$ is the number of samples after redundant averaging (and possibly also LST-stacking). This produces an inverse noise variance weight in the fit, where the noise on flagged channels is set to infinity.

In feathered inpaint, we use a logistic relative weighting that starts low and increases to 1 as we get far from any unflagged channels. Mathematically, that looks like:

$$W_{ij}^{\text{feather}}(\nu) \propto \frac{N_{ij}(\nu)}{V_{ii}^2}\left[\xi_{ij}(\nu) + (1-\xi_{ij}(\nu))\left(1+\exp\left[-\ln(\zeta_{\text{ip}}^{-1}-1)\frac{d_{ij}(\nu)-\ell_{\text{ip}}}{\ell_{\text{ip}}}\right]\right)^{-1}\right] \tag{3}$$

where $d_{ij}(\nu)$ is the number of channels to the nearest unflagged channel and where ℓ_{ip}, defined as

$$\ell_{\text{ip}} \equiv \frac{f_{\text{ip}}}{\tau_{\text{ip}}\Delta\nu} \tag{4}$$

is a characteristic unitless length scale in number of channels (of width $\Delta\nu$) beyond which one might worry about pop-ups.

The three parameters that control this feathering are:

1. The maximum delay out to which inpainting is to be performed, $\tau_{\rm ip}$. In practice, DPSS modes can go out a bit further, depending on the eigenvalue cutoff (Bull, 2024; Ewall-Wice et al., 2021). Larger delays capture more real spectral structure from foregrounds and other systematics by introducing more degrees of freedom into the fit. This makes them more prone to pop-ups in relatively small gaps in traditional inpainting. In this IDR, we use $\tau_{\rm ip} = 1000\,{\rm ns}$, which is more aggressive than the 500 ns used in HERA Collaboration et al. (2025).

2. The zero-distance weight, $\zeta_{\rm ip}$. When $d_{ij}(\nu)$ is zero, the exponential reduces to $\zeta_{\rm ip}^{-1} - 1$, which means that the relative weight of such a channel (compared to an unflagged channel) would be $\zeta_{\rm ip}$. This is a bit fictional, since when $d_{ij}(\nu)$ is zero, the channel is unflagged by definition. Nevertheless, $\zeta_{\rm ip}$ is useful to think of as a parameter setting how low the weight can go. In this IDR, we use $\zeta_{\rm ip} = 10^{-2}$.

3. The inpaint width factor, $f_{\rm ip}$. For a fixed length scale in number of channels set by $\tau_{\rm ip}$ and $\Delta\nu$, this factor sets the width of the logistic curve and thus how quickly it approaches 1. In this IDR, we use $f_{\rm ip} = 0.5$, which somewhat aggressively avoids pop-ups by approaching 1 fairly quickly with a characteristic width of $\ell_{\rm ip} = 4.096$ channels.

All of these parameter choices could be revisited (or modified for re-inpainting during LST-stacking—see subsection 4.2), though doing so for the per-night inpaint would be expensive.

3.3.3 2D-Informed Inpainting with Feathering

What we have not yet addressed in our discussion of feathered inpainting in subsubsection 3.3.2 above is the question of what to use for $V_{ij}^{\rm scaffold}(\nu)$ in Equation 1. One obvious candidate is to use an average of other nights, hoping they have unflagged data where we'd like to inpaint. This is sometimes the case, though not always, and it's an idea we revisit in subsection 4.2. Another candidate is a set of simulated visibilities, though we can quickly reject this given the relatively poor match between our visibilities (especially on short baselines) and our `abscal` model.

Instead, we have devised a rather complicated (but apparently effective) way to create a $V_{ij}^{\rm scaffold}(\nu)$ on a per-day basis *ex nihilo*, without using other days or an external model. We do this using 2D DPSS fits to the data, just as we did with autocorrelations in round 1 of RFI flagging, with calibration solutions in `smooth_cal`, and in round 3 of RFI flagging with highly-redundant corner-turned visibilities (see subsubsection 3.3.1). The process is, in fact, most analogous to the last step, which featured a relatively high delay filter combined with a fringe-rate filter that was designed to encompass all sky-like fringe-rates. That this is possible at all is due to recent updates to `hera_filters` that sped up fitting with a separable 2D basis.

However, in the case of RFI flagging, we don't care about pop-ups; we are merely trying to remove as much sky-like structure in the data as possible. What the fit does in the flagged regions doesn't matter, since they are already flagged. In inpainting, that is not the case! While a 2D fit can help us form a reasonably sky-like $V_{ij}^{\rm scaffold}(\nu)$ in large flag gaps when nearby times do not have similar gaps, it can still pop-up in large 2D flagging gaps. So we instead take a cautious, iterative, and annoyingly computationally expensive approach.

First we perform a 2D DPSS fit to the data we have (with zero weight to flagged data) that includes all sky-like fringe rates, but only goes out to 25 ns. We use this to inpaint the missing data,[2] and fit the inpainted result with 2D DPSS modes out to 50 ns, treating the inpainted data with exactly the same inverse noise variance weighting that we apply to the data (autocorrelations are separately inpainted to 100 ns to enable this). We use that fit to *re-inpaint*, then iterate: increasing by 25 ns each time until we get to 1000 ns. This process is expensive, but much more stable than a one-shot inpainting. In our experience, pop-ups are quite rare and are usually the consequence of large flagging gaps in integrations surrounded in time by many

[2]For iterative formation of the 2D DPSS scaffold is done without feathering. It's not entirely clear how to do feathering in a 2D fit, though we experimented with it. It seems that the slow, caution iteration is sufficient to "forget" the bad initial fits as we iterate. That said, perhaps there is a more clever way to come up with the 2D DPSS scaffold that is faster and/or more robust. We make no claim to the optimality of the algorithm; it is in fact, quite slow.

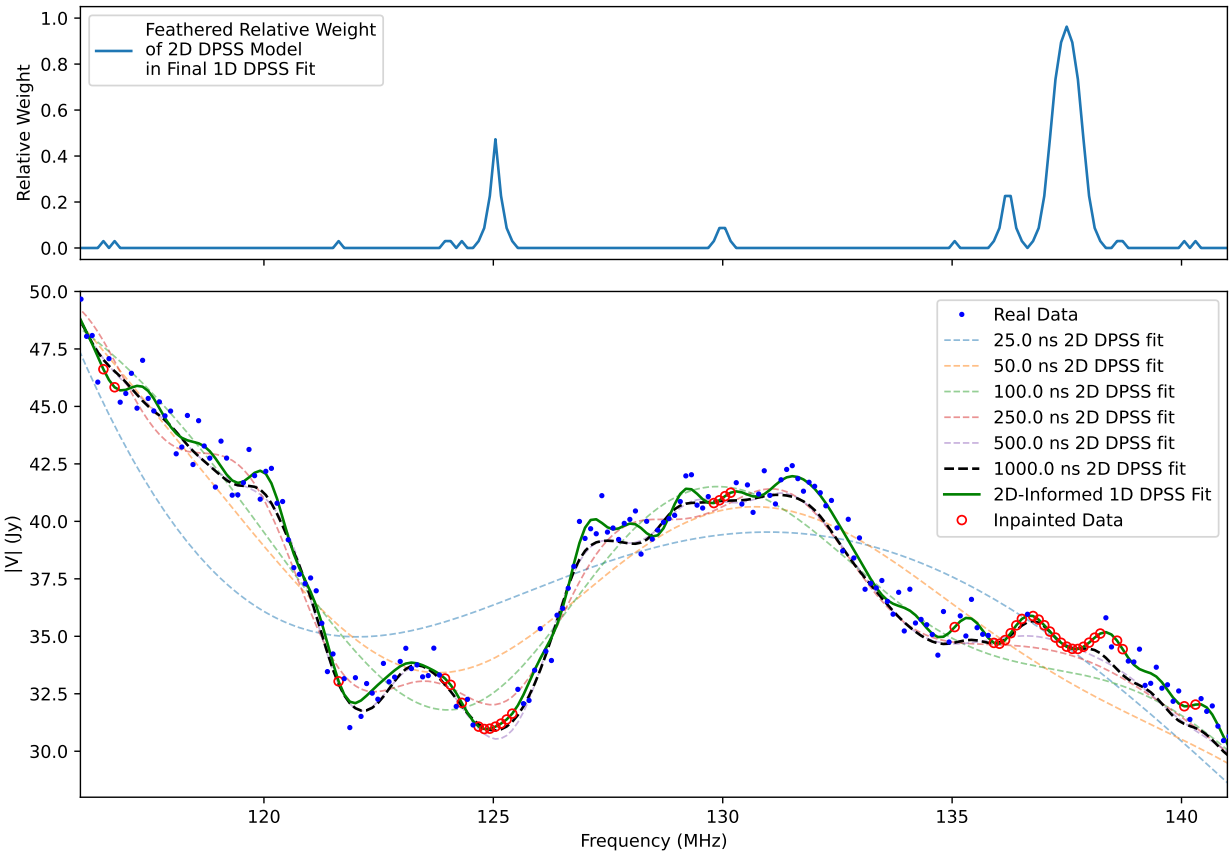

Figure 5: In this demonstration of 2D-informed feathered inpainting, we show the steps taken to perform inpainting on a single integration at JD= 2459863.53555. First, we perform a series of 2D DPSS fits to the entire day, starting with a fit up to 25 ns and increasing by 25 ns until we get to 1000 ns. Each fit fills in the gaps with the result of previous fits to avoid 2D pop-ups. A subset of these intermediate fits are shown with dashed lines. The final one, at 1000 ns (black dashed line) is used as the scaffolding visibility for the 1D feathered fit to this particular integration. The relative weights assigned to the scaffold, shown in the top panel, are 0 wherever we have real data and increase logistically to 1 in large gaps. The final 2D-informed 1D DPSS fit (solid green line), is a fit to the data (blue points) where we have it, and the black dashed line where we do, with weights that are small near real data to avoid spectral structure when we fill in the gaps of the data with inpainted data (red open circles), which by construction match the solid green line.

completely flagged integrations. Completely flagged integrations are not inpainted. This process is done separately above and below FM.

Once a 2D DPSS fit is produced in this way, we can use it as our scaffold for feathered inpainting, a process we call 2D-informed feathered inpainting. This is also performed separately above and below the FM band. The results have been quite encouraging and pop-ups have been largely eliminated (though not entirely, a problem we address during LST-stacking; see subsection 4.2 below.) In Figure Figure 5, we show both the iterative construction of the 2D DPSS scaffold, the feathered weights applied to that scaffold, and the resulting inpainted visibilities. In Figure Figure 6, we show the technique applied to an entire night's visibility waterfalls for a single baseline in both instrumental polarizations. Despite some rather large gaps in the data in the first and third columns, it is difficult to distinguish inpainted data from real data by eye in the second (phase) and fourth (amplitude) columns.

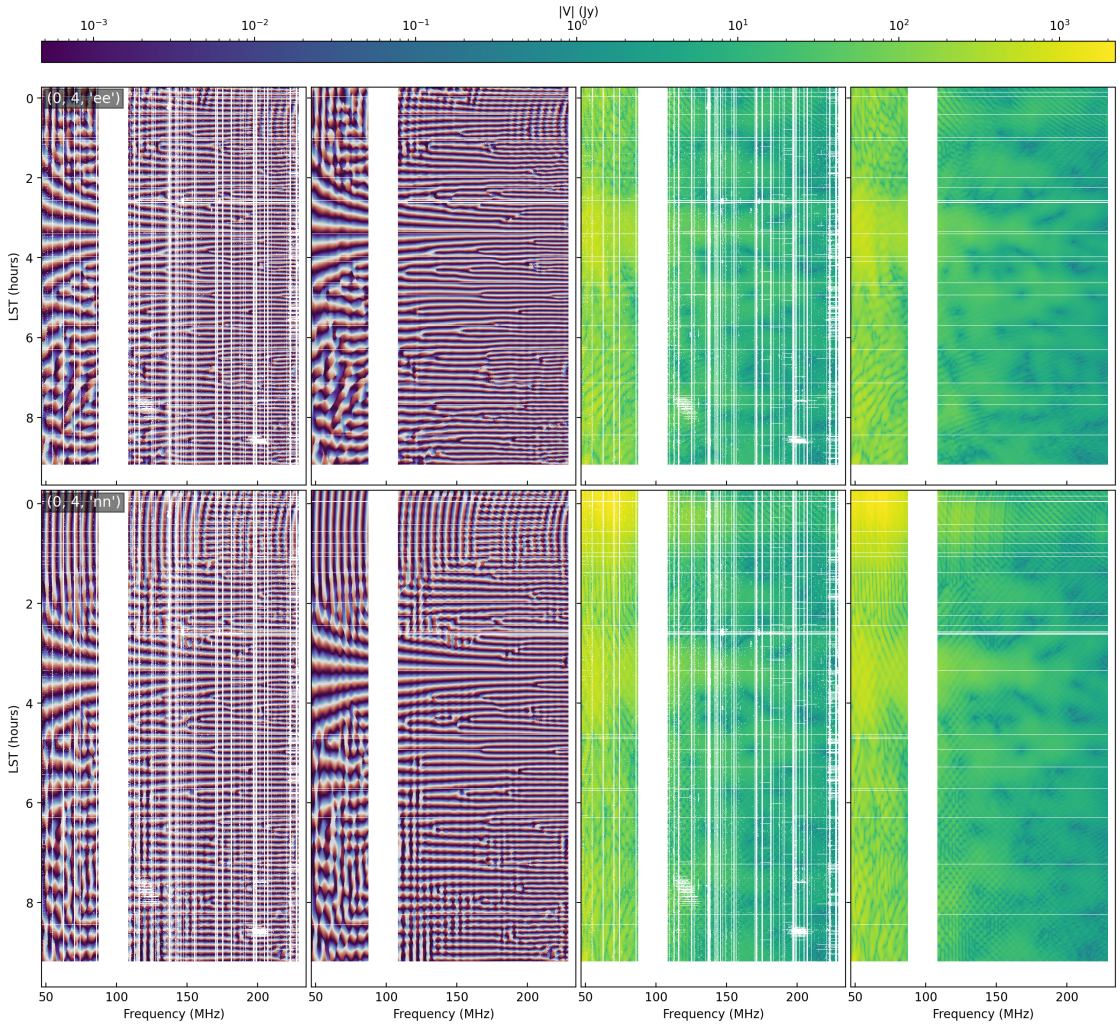

Figure 6: Here we show example results of 2D-informed inpainting with feathering on the 4-unit east-west baseline in both instrumental polarizations on 2459908. In the left two columns, we show visibility phase (linear, with a cyclic colormap). In the right two columns, we show amplitude. The odd columns show the flags in white, the even columns show the result after inpainting. Inpainting is performed separately above and below FM. The FM gap is never inpainted, nor are any integrations that are entirely flagged. The result is a set of inpainted visibilities that are difficult to distinguish by eye from real data.

3.3.4 Round 4 RFI Flagging: Phased Searches for High-Delay Structure

After the above new flagging and inpainting, we had originally assumed that we were ready to move to LST-stacking (see section 4). However, when we did that and started forming power spectra, we found clear excesses above the noise floor in certain bands. This was eventually traced back to specific nights and channels, visible clearly on even individual baselines after fringe-rate filtering to only keep modes that fringe like celestial sources in the main beam.

Such high-delay but apparently astrophysical sources are particularly worrisome: they weren't bright enough to show up in Round 2's incoherent search, but they mostly fringe like the foregrounds. They don't show up in Round 3 either, since that only looks for non-sky-like fringing. We're really seeing structure on a nightly basis that's nominally in the cosmologically interesting region of Fourier space, problematizing the wedge-window paradigm.

This investigation led us down quite the rabbit hole, eventually discovering an important new systematic—pulsar scintillation—and developing new tools to combat fringing RFI.

We begin by taking a very similar approach to Round 2 flagging. We pick the best observed baselines[3] (here those at least 10% as redundant as the autocorrelation, typically ∼150 total), divide by the noise predicted from the autocorrelations to form an SNR, and then high-pass delay filter them at 750 ns. Example results are in the `single_baseline_pI_SNR` notebook. However, instead of combining these results incoherently, as we did in Round 2, we instead combine them coherently at a fixed set of candidate sources, all of which are either bright pulsars or A-Team sources. These are:

- PSR J0628-28
- PSR J0742-2822
- PSR J0835-4510 (Vela pulsar)
- Fornax A
- Pictor A
- Crab
- Sagittarius A
- Centaurus A
- Cygnus A
- Cassiopeia A

Some of these we have clear evidence for signal, others are included speculatively out of an abundance of caution.

For each source we rephase[4] to, we produce a single z-score waterfall taking into account the mean and expectation value of the Rayleigh-distributed coherent sum. We restrict this calculation to a time range centered around the source's transit. In Figure 7, we show the results for a single interesting night, 2459935, for PSR J0628-28 and Fornax A.

Most striking, of course, is the result for PSR J0628-28. We are seeing clear evidence for refractive scintillation in the ISM on time and frequency scales that produce substantial spectral structure on a single night. Precisely why this pulsar (and not the even brighter PSR J0742-2822, which barely exhibits this behavior) shows up so strongly in this metric remains a question under active investigation. We also do not know why the effect diminishes so much below FM. Likewise, since we know the origin of the structure, it may be that the emissions can be modeled and removed with relatively minimal signal loss, rather than simply flagging every bright feature in this waterfall. That possibility is also under active investigation.

We also see that certain bright sources, most notably Fornax A, but also likely Pictor A and Centaurus A, that show "cloud" or "haze" of high SNR points, which appears stronger below FM than above it. The stronger and longer-duration effect at low frequency might be explained by the widening beam at low frequencies, combined with the fact that Fornax never gets closer than about 6° from zenith. We don't expect the same sort of ISM scintillation that we get from the pulsars. The origin of this is unclear. One speculative hypothesis is interplanetary or ionospheric scintillation, though that needs follow-up.

Regardless of the origin of these effects, we take the conservative approach for the time being of simply flagging waterfall pixels across all baselines. This is done in an iterative style comparable to Round 2 and Round 3 flagging. For each source, we flag $>4\sigma$ pixels, $>1\sigma$ median channels or integrations, $>2\sigma$ watershed

[3] We have observed that significant high delay structure can detected at this stage at fringe-rate 0. In order to avoid it contaminating both this metric and the one in Round 5, baselines whose main beams overlap FR=0 were excluded from this calculation. Subsequently, we have added in the ability (developed for Round 6) to include those baselines while filtering out the FR=0 mode and doing an appropriate leverage correction for the lost signal. This was not done in the current run and the capability still needs testing. In particular, an open question remains as to whether the FR=0 filtering should happen only prior to Round 5 or prior to both Round 4 and 5.

[4] It is worth noting, since it was the source of some confusion during this investigation, that calibrated HERA data is not phased to zenith but rather "unprojected" in the parlance of `pyuvdata`. This means that rephasing requires simply taking out the geometric delay of the source for each baseline, no need to "put zenith back in," as the discussion around Equation 4 of Keating et al. (2024) clarifies.

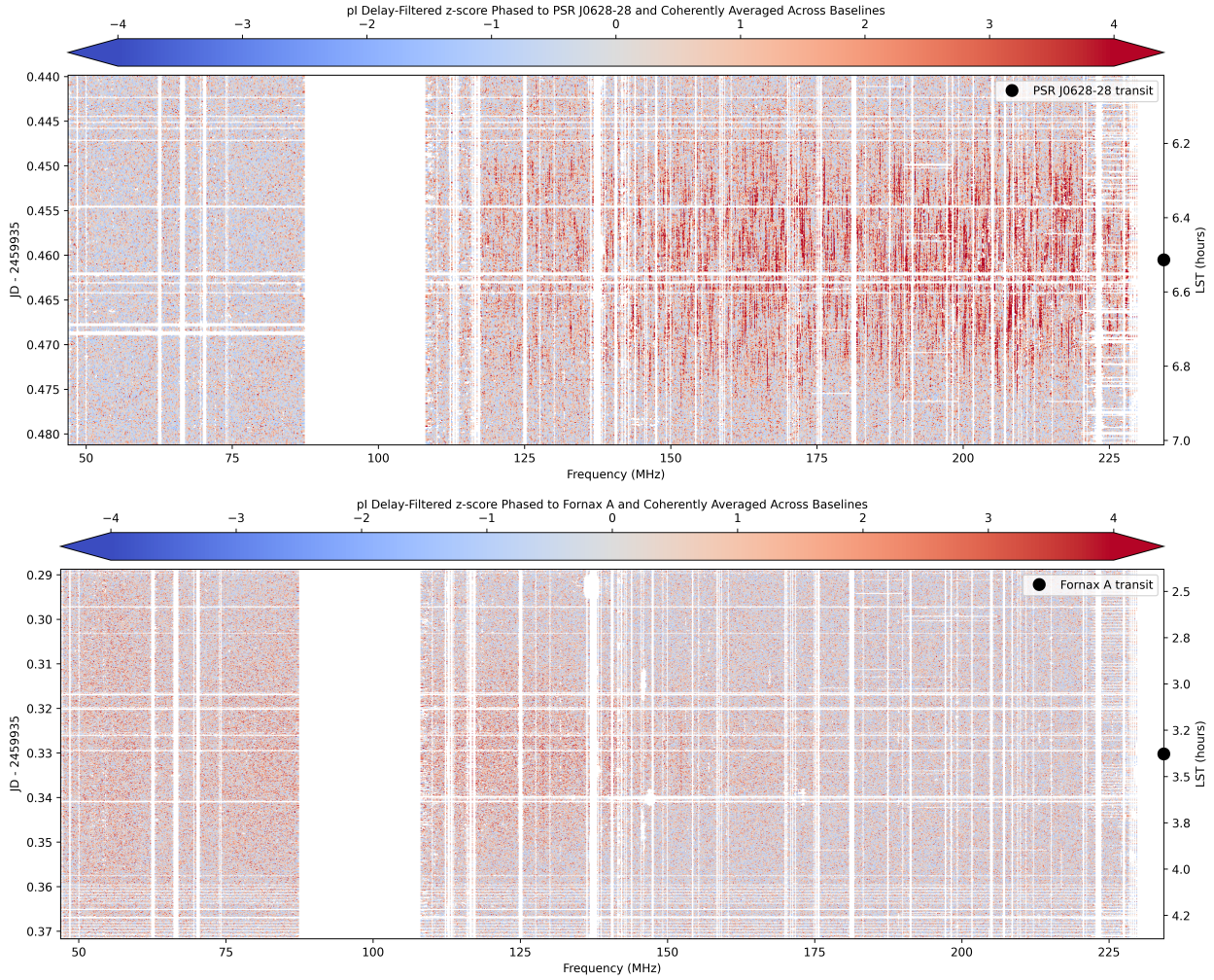

Figure 7: The input to the fourth round of RFI flagging are delay-filtered and rephased SNRs, combined coherently at several point sources. Here we show the results for 2459935, a night with some particularly bad new RFI. PSR J0628-28 (top panel) is by far the clearest signal, and appears the result of refractive scintillation. However, extragalactic sources like Fornax A also seem to be showing a different sort of structure, both apparent spectral lines as well as a "cloud" or "haze" of high SNR points, more clearly below FM. While it has been speculated that this is due to interplanetary or ionospheric scintillation, that hypothesis merits further investigation.

neighbors to previously flagged channels, then we do iterative time-convolved flagging up to 6 minute scales (a temporal analog to frequency-convolved flagging used on Round 3; see subsubsection 3.3.1), then $>2\sigma$ watershed again, then $>1\sigma$ mean channels or integrations, and then one final watershed.

With the exception of whole channel flags, the above flags are all restricted to a fixed range in LST around the source's transit. The flags are then saved separately, but ORed together in future steps. An example result for 2459935 is shown in Figure 8. Results for the rest can be found in the `full_day_rfi_round_4` notebooks.

3.3.5 Round 5 RFI Flagging: Phased and Incoherent Searches for Fringing High-Delay Structure

Round 5 RFI flagging is identical in structure and reuses much of the code infrastructure as Round 4. The key difference is that Round 5 uses delay- and main-beam fringe-rate filtered SNRs. This is the original metric

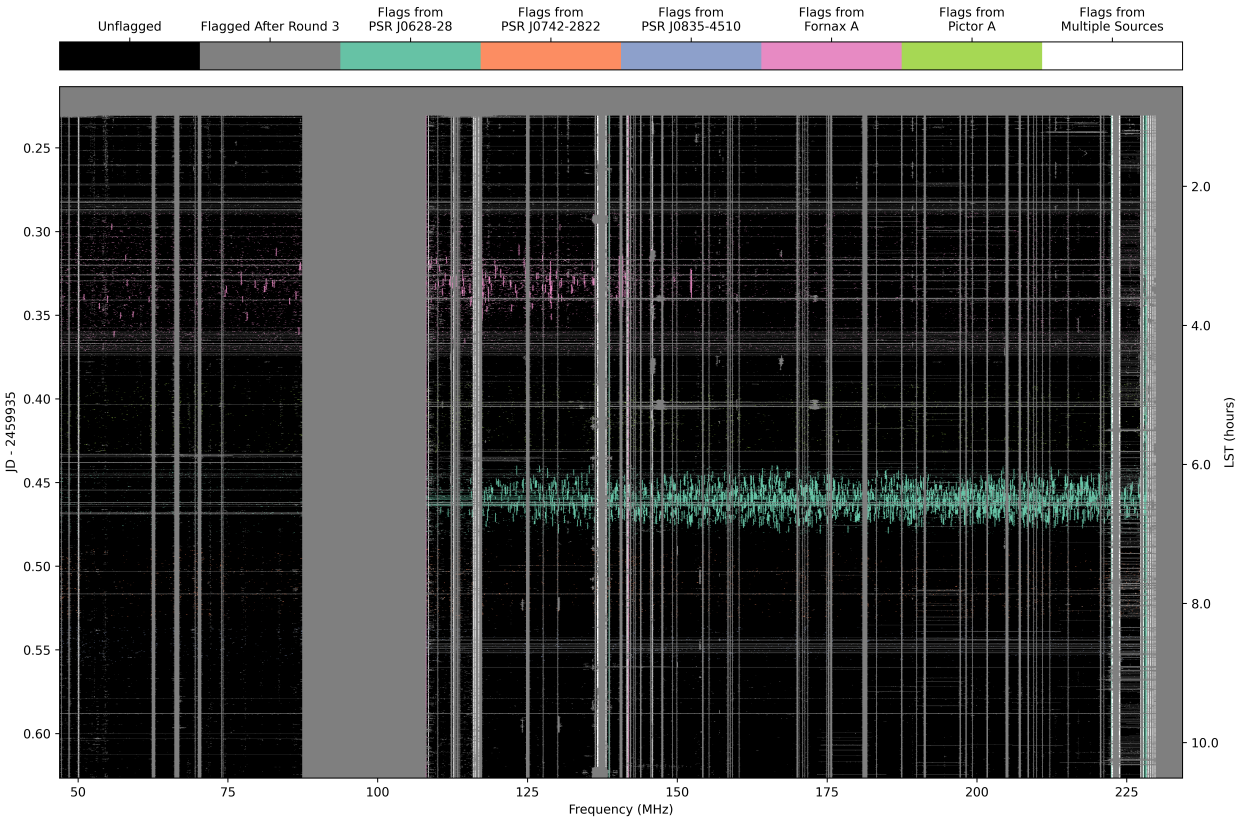

Figure 8: Flagging summary after the fourth round of RFI flagging on 2459935. Similar to Figure 4, but here generated from SNRs rephased to particular known pulsars and radio galaxies. The final flags are ORed together, though flags from individual sources can only affect a relatively compact range of LSTs around the transit of that source.

in which this fringing RFI was discovered. It is also a space in which we need to be a bit more careful about signal loss, since it's precisely where we intend to estimate the 21 cm power spectrum. Strictly speaking, including Round 4 as described above is unnecessary and we could skip right to Round 5. However, because the fringe-rate filter spreads out RFI that's fringing but somewhat compact in time (like pulsar scintillation), we suspect that performing Round 4 without the fringe-rate filter and Round 5 with it is more surgical.

When we coherently combine delay- and main-beam fringe-rate filtered SNRs using the Round 4 flags as a prior, we see that we missed quite a bit of structure. Figure Figure 9 shows this for both PSR J0742-2822 as well as Fornax A. The extra structure in the pulsar makes sense: it's low-level scintillation missed by Round 4 and then spread out by the fringe-rate filter (diluting the signal, but not as much as the fringe-rate filter suppresses the noise). The spectral structure on Fornax is harder to understand and seems to have a fundamentally different origin from the speckle or haze seen in Round 4 (see Figure 7).

In addition to flagging on regions near the transit of a fixed set of sources, we also flag on the incoherent average filtered SNRs. This is directly analogous to Rounds 2 and 3, except here we use a substantially more sensitive data product. Interestingly, we see phenomena that are not clearly associated with any of the sources we've investigated thus far. We show the results for 2459935 in Figure 10 and for 2459860 in Figure 11 to demonstrate some different phenomena. The origin of these blips, some long and some short, remain unclear. Some may be real spectral structure, some may be filtering artifacts. Further investigation is warranted.

Both coherent and incoherent waterfalls are used for flagging. However, due to concerns about overflagging, the thresholds set in subsubsection 3.3.5 were all doubled and the timescale for iterative time-convolved flagging was increased to 1 hour. The result for 2459935 is shown in Figure 12. Results for the rest can be

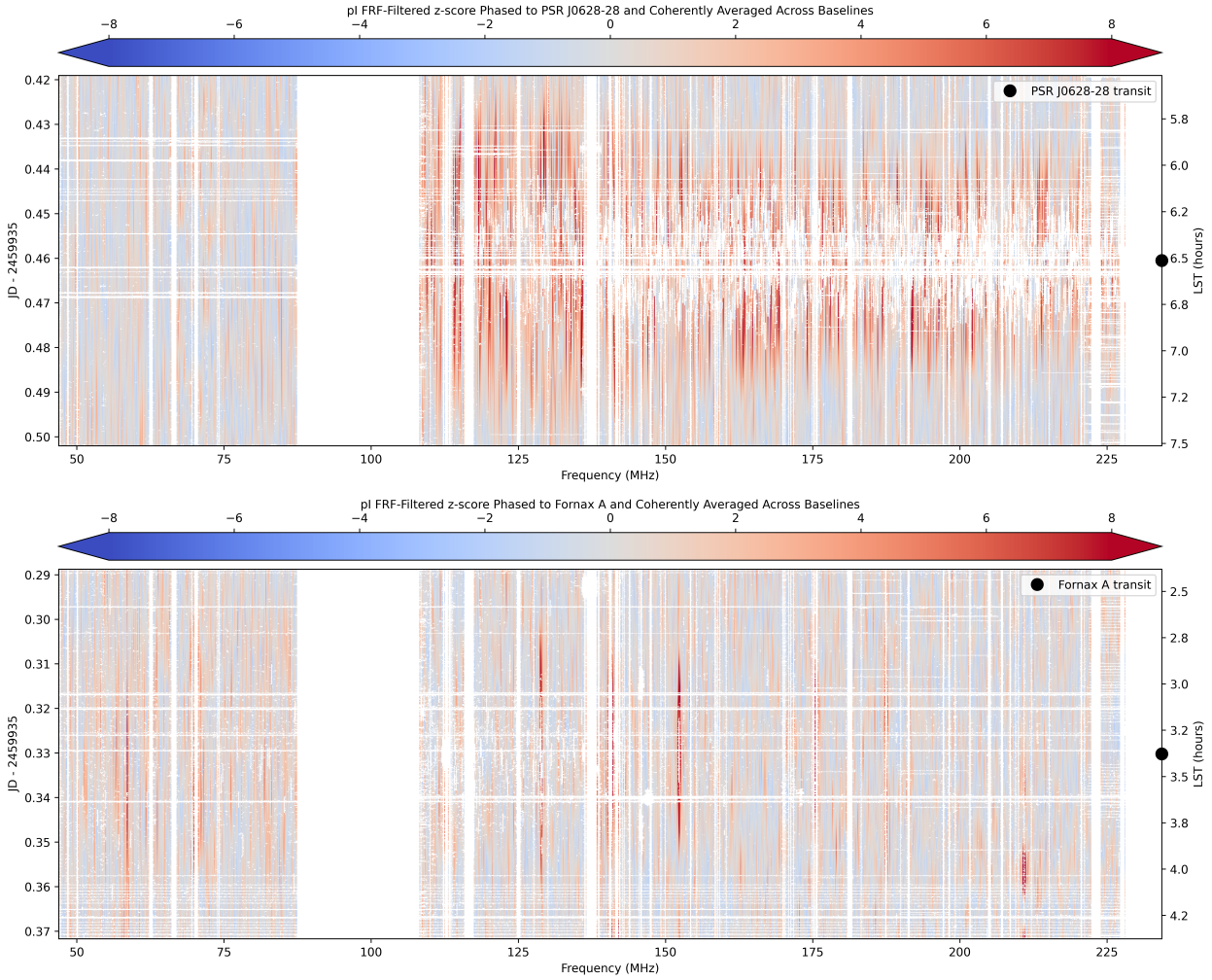

Figure 9: Similar to Figure 7, except here a main-beam fringe-rate-filter has been applied to individual delay-filtered SNRs before coherently combining across baselines. Top panel is PSR J0742-2822, bottom panel is Fornax A, both observed on 2459935. Prior flags come from Round 4. These contribute to the discovery of additional flags for Round 5, along with the incoherent average shown in Figure 10.

found in the `full_day_rfi_round_5` notebooks.

Finally, it should be noted that five rounds of RFI flagging certainly seems excessive! It may be possible to combine some of these in future analyses (2 and 4 seem like the clearest candidates for combination). The challenge to overcome in such a simplification is that the structure we want to remove impacts the filtered data products, leading to under- or over-flagging of neighboring times and channels. And while filtering can be parallelized over times or baselines, the flagging decision cannot, yielding a bottleneck.

3.3.6 Re-Inpainting

With the new flags generated in Rounds 4 and 5, we have more times and frequencies that need inpainting. The original 2D-informed inpainting (see subsubsection 3.3.3) is the most computationally expensive step in the pipeline. However, for re-inpainting we can circumvent the most expensive step (iteratively generating the 2D DPSS fit) by simply using the previous inpainted data as the scaffold for the 1D feathered fit. So we use the exact same equation and parameters for feathering presented in subsubsection 3.3.2, but it's roughly a factor of 10 to 20 times faster. Results for one baseline per night can be inspected in the `single_baseline_scaffolded_and_feathered_inpainter` notebooks.

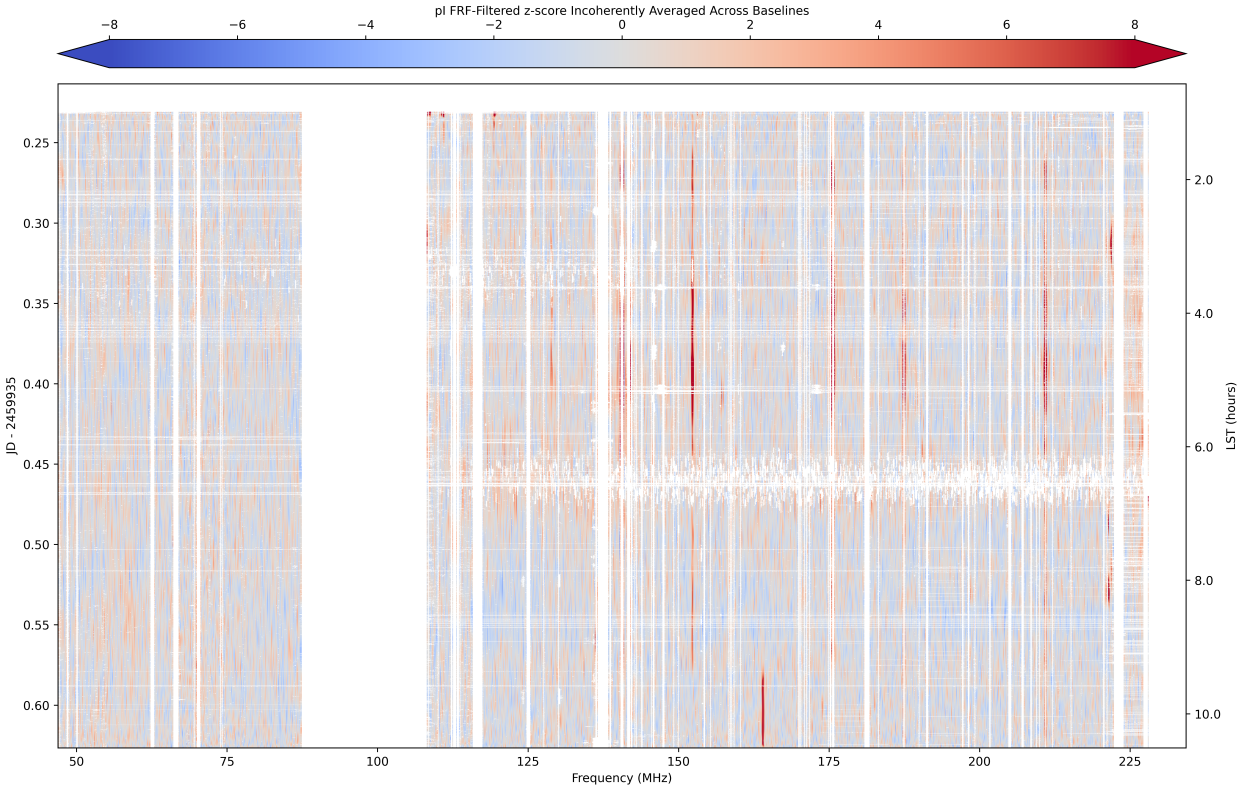

Figure 10: Here we show the results of incoherently averaging delay- and main-beam fringe-rate filtered SNRs on 2459935, one of the metrics used for Round 5 flagging. While coherent averaging is more sensitive to emission from individual sources, incoherent averaging reveals an overlapping set of phenomena. For example, the strong blip at 154 MHz seen in Figure 9 extends for a much wider range of LSTs. Other features do not appear to correspond to individual sources. The nature of this structure remains unclear. However, we do know that it changes dramatically from day to day. To what extent it is attributable to individual baselines or all baselines remains to be investigated.

In future analyses, using an "external" scaffold is a potential way to vastly speed up the pipeline for both H6C re-analyses as well as H7-9C analyses. Here, doing so is easy, since the data product matches the unflagged data and is smooth by construction. In the future, we plan to use processed (perhaps delay- or fringe-rate-filtered, or both), LST-stacked data as the scaffold. This will be fast, but it presents a new challenge: what happens if the scaffold doesn't quite match the data? Feathering should help prevent the introduction of spectral structure. However, we may need to explore schemes by which the LST-stacked visibilities may be "uncalibrated" in a relatively smooth way to match the individual day, so that inpainting imparts a minimum of spectral structure. This remains an open question for future work.

3.4 Parameter Tweaks

In addition to the new analysis steps and algorithmic changes, we list here for completeness a couple of other parameter tweaks for this IDR.

Prior to redundant calibration, antennas are thrown out for correlating poorly (often due to clock issues) using the correlation metric defined in (Storer et al., 2022). Previously we used 0.2 as the threshold for flagging (and anything under 0.4 as suspect). We raised this cutoff to 0.35 (and 0.45) upon closer inspection of the distribution of the statistic over days and evidence that a few low-correlating antennas were getting through.

In Round 2 RFI flagging, we tweaked various thresholds in response to some work we did to flatten out the z-scores we used for flagging using a more exact calculation of the leverage (Dillon and Cox, 2024).

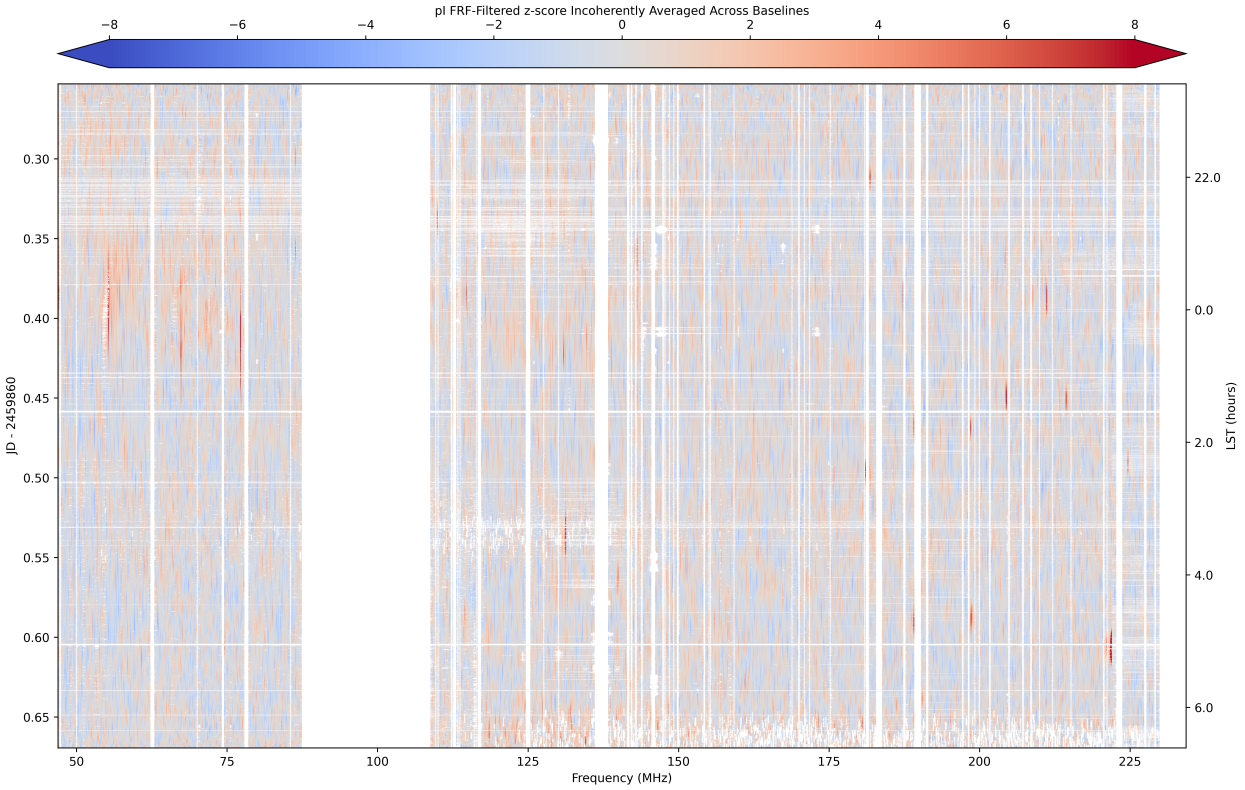

Figure 11: The same as Figure 10, except for 2459860 instead of 2459935. We include this just to highlight the diversity of phenomena that show up at this stage, especially the shorter blips which show up on many days at different times and frequencies.

Additionally, the `max_time_flag_frac` parameter was raised from 0.1 to 0.25, since we can now rely on an improved ability to flag broadband (but time variable) RFI in Round 3. We also added iterative time-convolved flagging (just as in Rounds 4 and 5) on time scales up to 5 minutes. This was developed to better flag high-delay structure in incoherently-averaged, delay filtered visibilities that we later identified as scintillation from PSR J0628-28 and were much better able to flag after rephasing to the pulsar.

4 LST-Stacking

Since we've already performed the corner turn and since power spectra are computed from individual redundantly-averaged baselines after fringe-rate filtering, it makes sense to stack baselines over nights rather than individual LST-bins over nights as has previously been our practice. This is made even easier by relabeling all redundantly averaged baselines after the corner turn to use a set of baseline keys that does not depend on which antennas are flagged. We have thus re-engineered our LST-stacking to work on single-baseline (but 4-polarization) files, one from each night.

As it stands, we first stack a subset of baselines, use them to perform per-night LST-cal (see subsection 4.1 below) to recalibrate redcal degeneracies, and then perform a final stacking. That stacking is described in subsection 4.2 below. The entire LST-stacking pipeline is controlled by a single TOML, analogous to that used in per-night analysis. However, instead of treating individual files as the "basename" unit, this new type of makeflow treats strings like `'0_1'` as the basename to parallelize over.

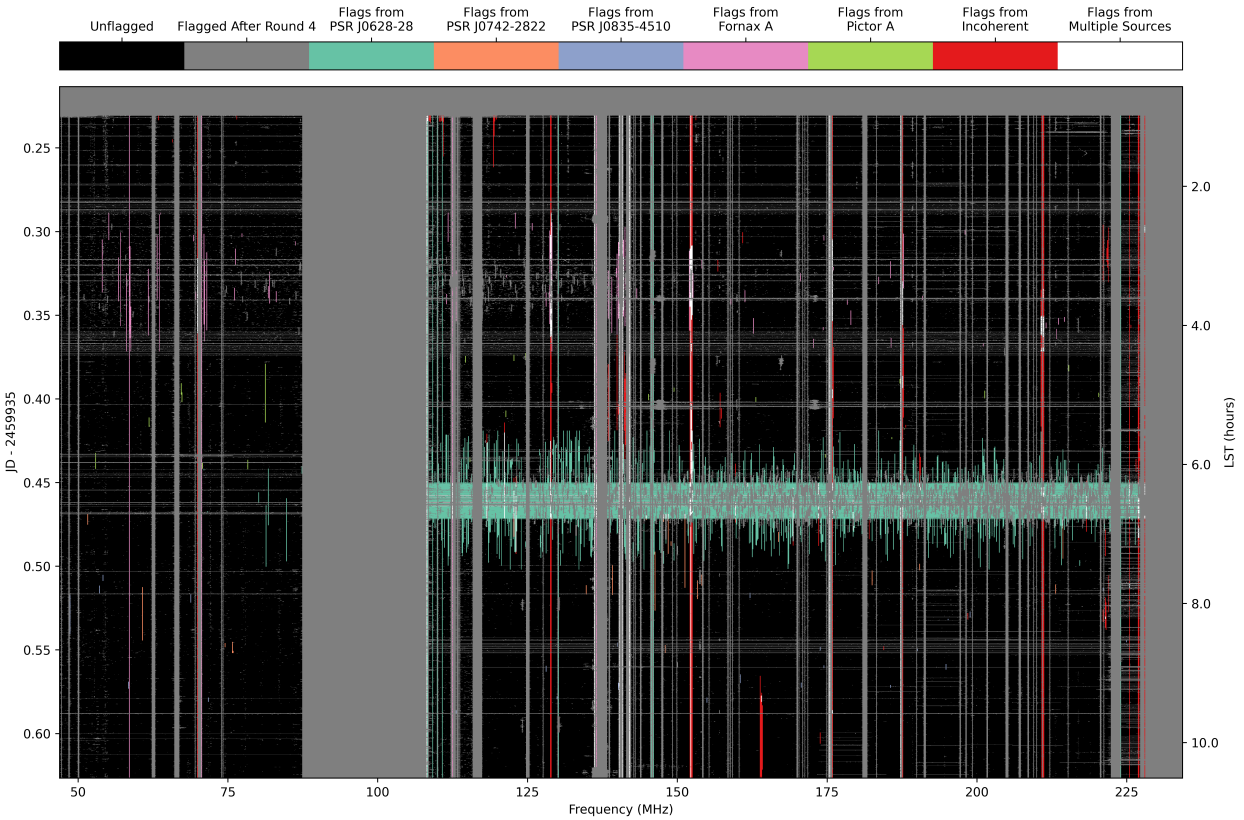

Figure 12: Flagging summary after the fifth and final (*hopefully!*) round of RFI flagging on 2459935. Similar to Figure 8, except this uses Round 4 as a prior and includes both filtered SNRs coherently averaged on individual sources as well as incoherently averaged. Note that both Rounds 4 and 5 flag integrations greater than 50% flagged, as often happens for PSR J0742-2822 with its extremely strong scintillation.

4.1 Updates to LST-Cal

Before performing LST stacking and averaging of the per-night, redundantly averaged visibilities, we apply an additional calibration step designed to improve the alignment of visibilities across nights that share a common LST bin. This step, referred to as *LST-binned calibration* (hereafter `lstcal`), adjusts the per-night redundant-calibration degeneracies in order to minimize the variance of visibilities across nights. The inclusion of `lstcal` is motivated by the H6C IDR2 analysis (see HERA Memo 131), which found that the variance of visibilities across nights exceeded the level expected from thermal noise alone. This excess variance is believed to arise primarily from residual calibration errors that remain after redundant calibration and gain smoothing. By solving for degeneracy parameters that minimize cross-night variance within each LST bin, `lstcal` reduces these residual differences and improves the consistency of visibilities across nights.

A major difference between the H6C IDR2 and IDR3 analyses is the organization of the per-night data products that serve as inputs to LST stacking and LST-binned calibration. In IDR2, the per-night pipeline produced files containing two integrations, all frequency channels, and all redundantly averaged baselines. In contrast, the IDR3 pipeline performs a corner turn prior to LST processing and outputs *single-baseline files* that contain all times, frequencies, and polarizations for a single redundantly averaged baseline. This restructuring was introduced to support improved RFI flagging and inpainting in the post–corner-turn analysis stage (as described in the previous sections). While this organization simplifies several downstream processing steps, it complicates the implementation of `lstcal`. In the IDR2 pipeline, all redundantly averaged baselines could be loaded into memory simultaneously, allowing `lstcal` to be performed directly on the full set of visibilities during LST binning. However, given the IDR3 file structure, where each night consists of many single-baseline files, loading the full set of visibilities from all nights simultaneously is no longer

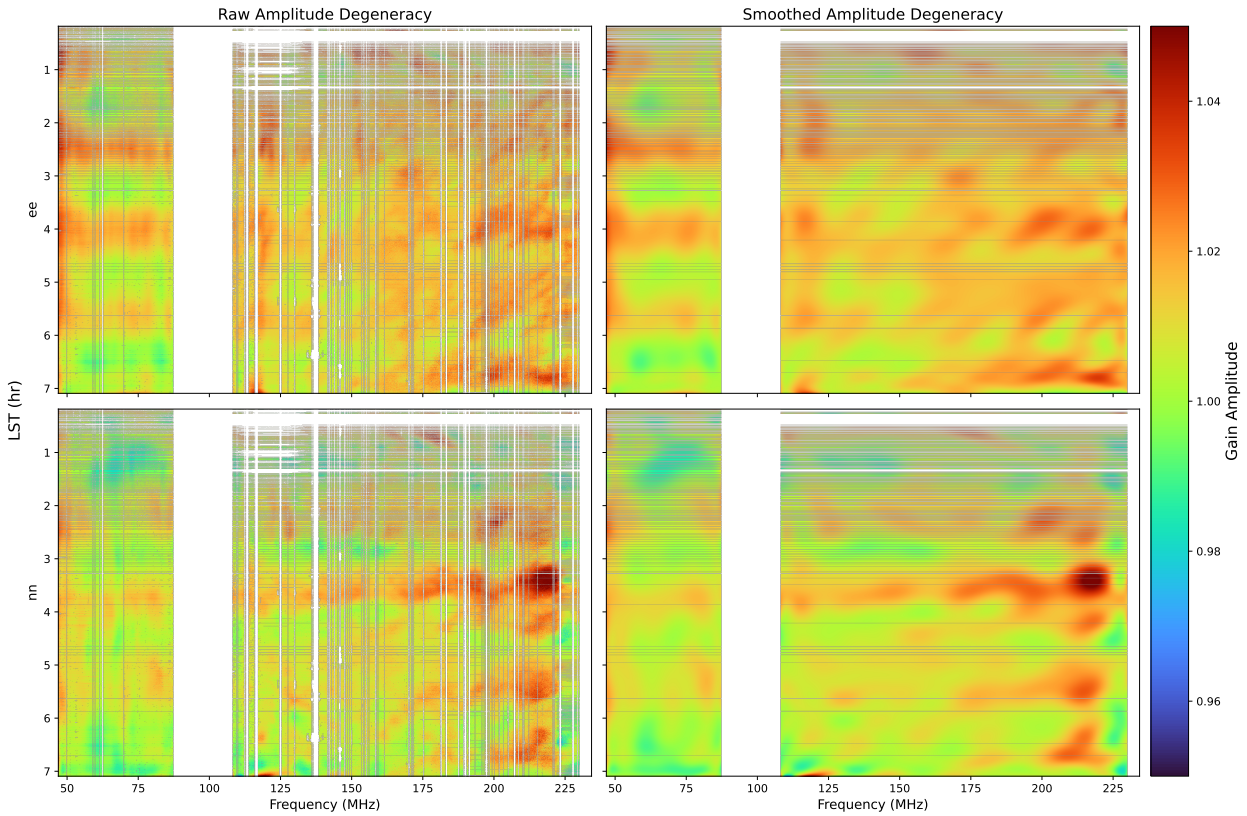

Figure 13: Example of the amplitude component of the LST-binned calibration (`lstcal`) gain degeneracy solutions for JD 2459869. The top and bottom panels show the two polarizations (`ee` and `nn`). In each panel, the left column shows the raw amplitude degeneracy solutions as a function of frequency and LST, while the right column shows the same solutions after smoothing in time and frequency.

feasible due to memory and I/O constraints. To accommodate this change while still performing LST-binned calibration, we modify `lstcal` to operate on a representative subset of redundant baselines selected to have the largest number of samples across nights.

As in IDR2, `lstcal` begins by computing a preliminary LST-binned average for each selected baseline. This averaging stage incorporates the per-night inpainting described in the previous section in order to minimize the impact of flagged data when constructing the reference visibility for each LST bin. The resulting nightly averages serve as a reference against which individual nights are calibrated.

For each night, we then solve for the redundant-calibration degeneracies by fitting that night's visibilities to the reference LST-binned average within each LST bin. These fits are performed independently for each frequency channel and time sample. Figure 13 shows an example of the resulting amplitude degeneracy solutions for a representative night. The resulting degeneracy solutions are subsequently smoothed in time and frequency to prevent introducing spectral structure into the calibration solutions. Finally, the smoothed degeneracy solutions are applied to the full set of per-night visibilities prior to the final LST-stacking stage. Figure 14 shows the effect of `lstcal` on the excess variance statistic, demonstrating that this step significantly reduces night-to-night scatter beyond what is expected from thermal noise alone. The excess variance statistic used in this comparison is

$$\gamma = \frac{\sum_{i=1}^{N} n_i (V_i - \bar{V})^2}{\sigma^2 (N-1)}, \tag{5}$$

where V_i denotes the visibility for night i, n_i is the number of contributing samples in that bin, $\bar{V}$ is the weighted mean visibility across nights, and σ is the estimated thermal noise level. Values of $\gamma > 1$ therefore indicate excess variance beyond that expected from thermal noise.

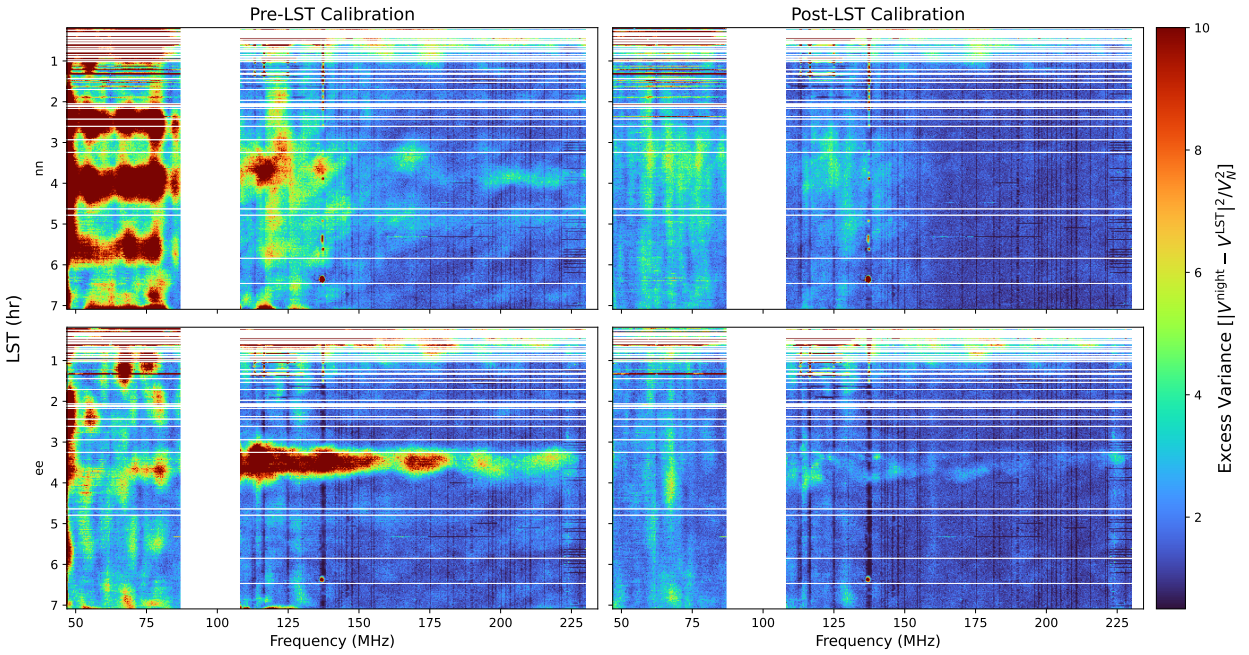

Figure 14: Comparison of excess-variance statistic for JD-2459869 before and after applying gains computed by LST-cal, averaged over the 30 redundant baseline groups with the largest number of N_{samples}. The left panels show the excess variance before LST-cal, revealing significant features across LST and frequency. The right panels illustrate the substantial reduction in excess variance after LST-cal gains are applied, especially in the low-band.

4.2 LST-Stacking with Re-Inpainting

In many ways, LST-stacking is more straightforward with corner-turned files. A lot of the bookkeeping is simplified. We can simply define a grid in LST, load in all baselines, rephase to the common grid, and stack. As in HERA Collaboration et al., 2025, inpainted data (which we keep track of with ancillary `where_inpainted.h5` files) are included in the average of nights, even though they do not count as samples when computing `nsamples` and subsequent noise statistics.

However, this stacking presents some new opportunities. The first is that it naturally reveals which nights are experiencing major inpainting failures. We simply compute the modified z-score over nights for each night, LST, and channel. For each LST, band (i.e., above or below FM), night, and polarization, if any visibility has a modified z-score greater than 5 and at least 3 nights go into calculating that z-score, we can decide to simply discard all inpainting that was done and re-do it. However, instead of using a 2D DPSS model as our scaffold, as we did in the per-night analysis (see subsubsection 3.3.3), we instead use *the other nights* as our scaffold for feathered inpainting.

We show a particularly dramatic example of this in Figure 15 using only the nights from IDR2 and an integration with a big flagging gap on 2459869. It was constrained from popping up too badly, but the per-night 2D DPSS fit must have had too little information from neighboring times to provide a good scaffold. This most often occurs when you have one or a few integrations with large gaps surrounded by several flagged integrations.

However, by using the average over all other nights that aren't identified as having outliers for that LST, band, and polarization as $V_{ij}^{\text{scaffold}}(\nu)$, we can use the same procedure described in subsubsection 3.3.2. In the example in Figure 15, five nights show outliers above the 5σ threshold somewhere above FM for that particular baseline, LST, and polarization. That leaves 9 other nights with which to form an average that other nights can be inpainted against. This process gets more robust as we include more and more nights in the stack. It should also be noted that although the per-night inpainting was the most expensive part of the pipeline, having *most* of the nights in agreement makes it much easier to find the ones that aren't. In

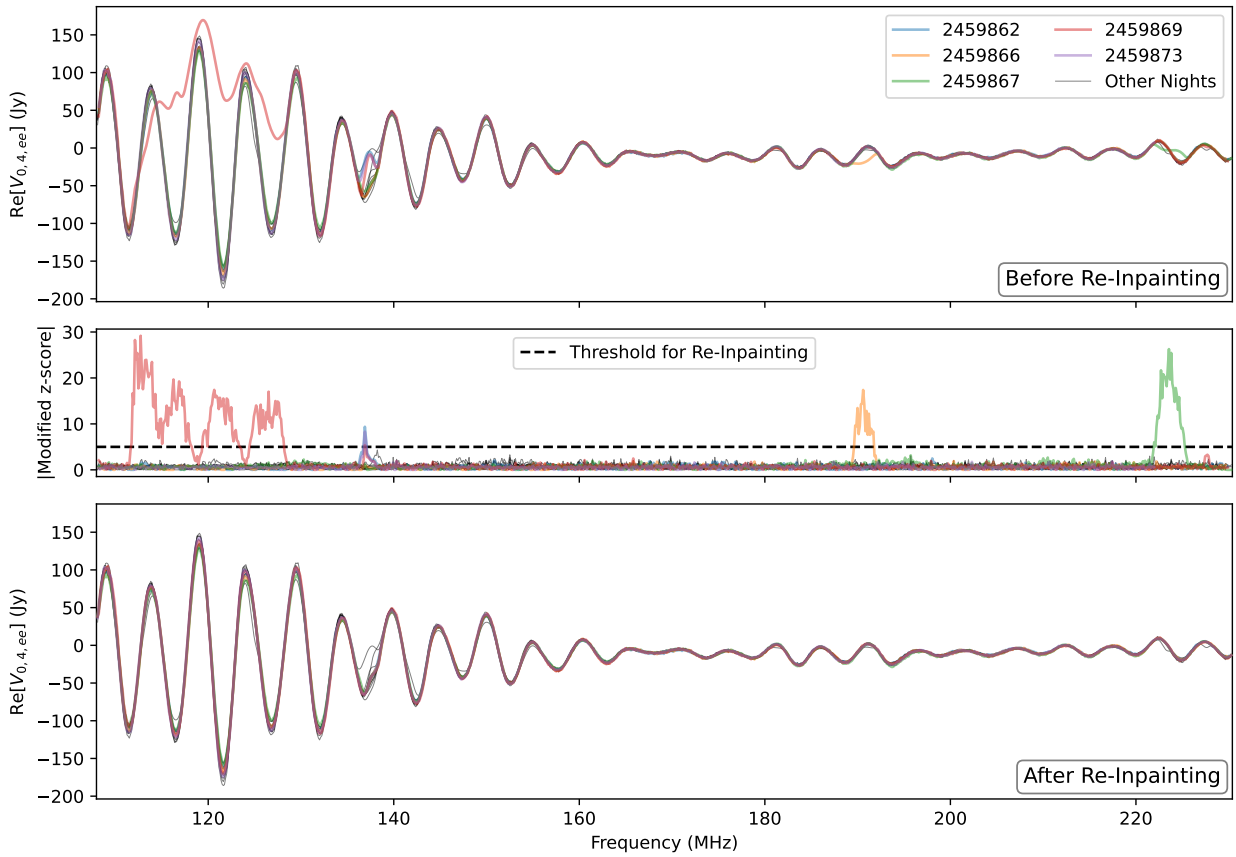

Figure 15: In this example of re-inpainting at 0.4832 hours LST for the 4-unit east-west, *ee*-polarized baseline, five nights of the 14 nights from IDR2 are outliers in at least one channel where per-night inpainting was done. As a result, *all* flagged channels for that night are reinpainted, using the average of non-outlier nights as the visibility scaffold for feathered inpainting, as described in subsubsection 3.3.2. Most of these discrepancies are clearly visible in the top panel, before re-inpainting. In the bottom panel, we show the result: data which are much more consistent across nights. Of note, when the variability across nights is high, as is the case for ORBCOMM channels (∼137 MHz), the degree to which any given night is considered an outlier goes down. The pop-up seen in 2459869 is a rather dramatic example of the problems even 2D-informed feathered inpainting can produce, albeit rarely.

Figure 16, we plot a 2D waterfall showing how much the maximum modified z-score is reduced on that same baseline, now looking at all 131 nights.

We can take this logic one step further. Different nights have different lowest and highest unflagged channels above and below FM. If we simply stack them together, we risk creating discontinuities. However, we can use the same method as above to "expaint" these data, averaging real data with extrapolated data scaffolded by that same real data. Practically speaking, it's unlikely that we will use expainted data. However, the process is cheap and it allows us to defer the decision of how and whether to divide up the channels into power spectrum sub-bands for as long as possible, supplying future analysis with maximum flexibility. The end result of this entire process is demonstrated in Figure 17.

Finally, the LST-stacking step is our final opportunity to perform additional per-night filtering, including filtering out the FR=0 mode. We see clear evidence of high delay structure at FR=0, which we ultimately filter out during power spectrum estimation (see subsection 5.3). However, variable time flagging from day to day means that LST-stacking will necessarily introduce some temporal structure, mixing the FR=0 modes with higher modes. Thus, after all re-inpainting and expainting is done, we also filter individual nights in time, removing modes with a DPSS fit with a half-width of 0.01 mHz. However, this feature should be

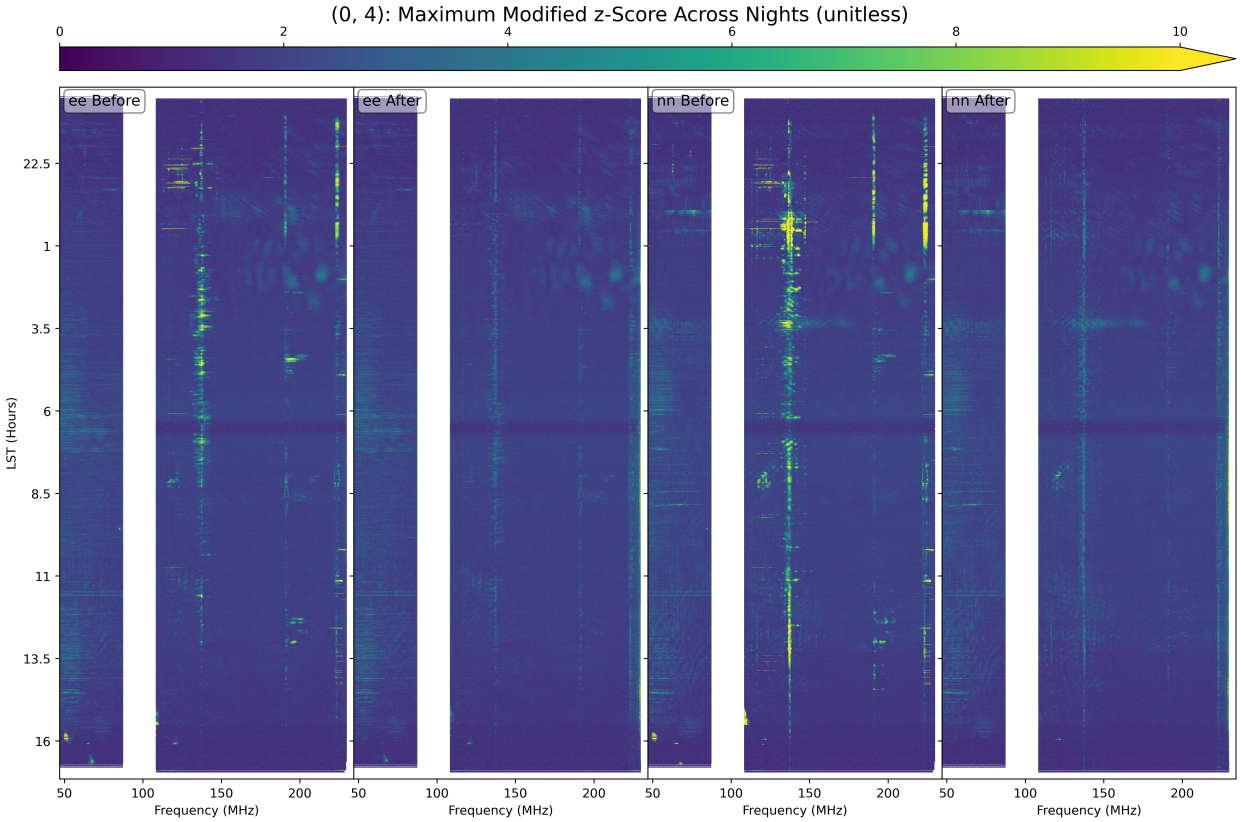

Figure 16: Re-inpainting allows us to identify the worst outliers in a given channel and LST that are attributable to inpainting failures (i.e. "pop-ups") and gives us a chance to correct them on a per-night basis prior to LST-stacking. Here we show the improvement in the single worst modified z-score across nights in each pixel of the waterfall, before and after re-inpainting. High z-scores are particularly common in heavily inpainted regions, like ORBCOMM at ∼137 MHz, and in individual 8 MHz TV allocations at high frequency.

considered experimental and we've seen some tentative evidence that it does not play nicely with expainting. This needs to be revisited before these data products can be used going forward. However, if they can be made to work reliably, they should help narrow the range of contaminated modes in the LST-stacked data that are truly FR=0 in individual days.

5 Post-LST-Stacking

Finally, we come to post-LST-stacking analysis. This last step in the pipeline is the newest, most speculative, and perhaps most subject to change before publication. This is in no small part due to the tremendous data reduction of LST-stacking, making iteration easier. In H6C IDR 2.3 (Dillon et al., 2024), we went right from LST stacking to power spectra (though we first had to do a corner turn since LST-stacking was done per-LST-bin and not per-baseline). Here, we've added an additional step to look for low-level RFI, flag it off, and then re-inpaint any newly flagged data, as well as any entirely inpainted data, using the output of the LST-stacker as a scaffold. This process has been incorporated into the `h6c_pspec_11band.toml` file responsible for specifying both the final round of flagging and re-inpainting as well as the single baseline power spectrum estimation.

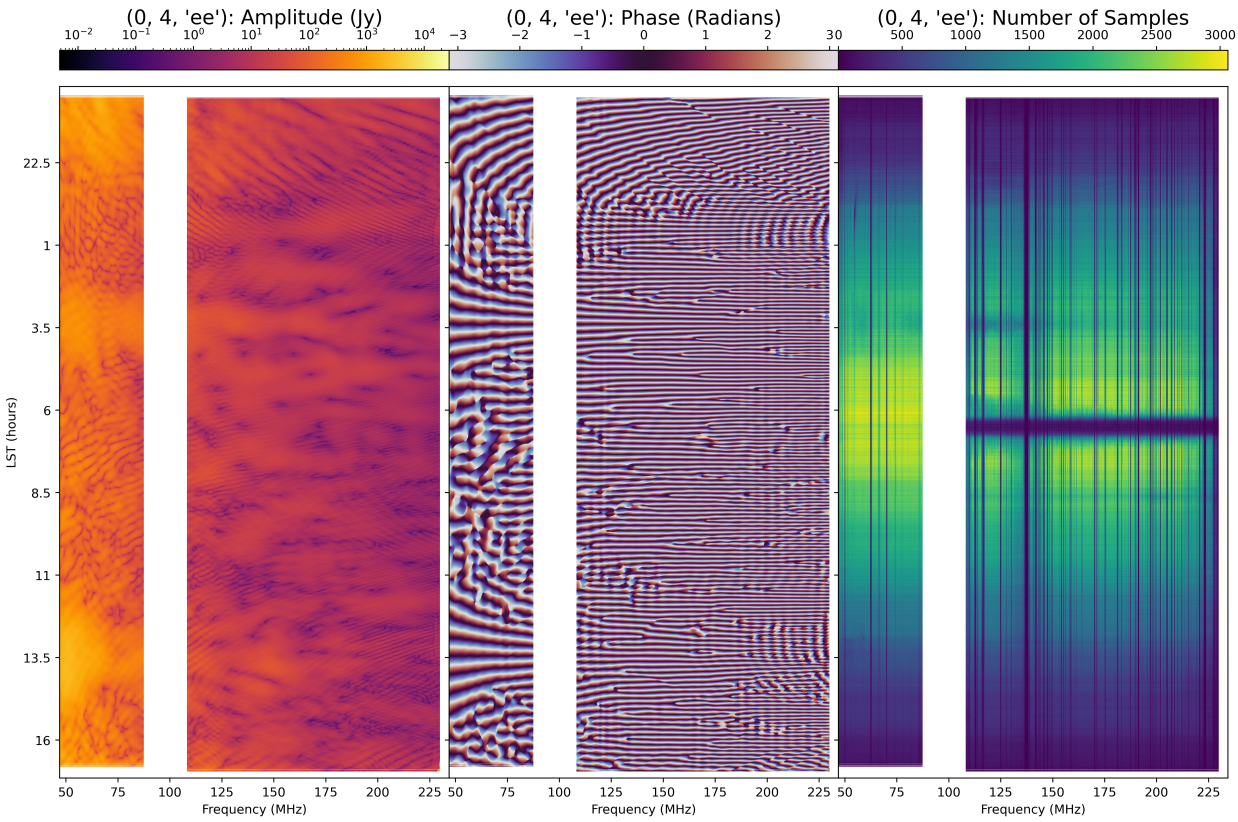

Figure 17: The final LST-stack for the 4-unit EW baseline. Despite all the inpainting, made clear by decrements in the number of samples visible in the leftmost panel, little obvious evidence of that inpainting persists in final stack. The big gap at ∼6.5 hrs LST is due to a huge fraction of data getting flagged for spectral structure due to the scintillation of PSR J0628-28 when it passes through the primary beam. More subtle is a tiny pop-up in the bottom right corner, which is due to poorly constrained expainting. Likely these LSTs (>16 hrs) should be ignored when forming power spectra anyway.

5.1 Final Round of RFI Identification

To look for RFI that slipped through rounds 4 and 5, we perform another round of delay- and fringe-rate filtering highly redundant baselines to produce per-baseline SNRs, and then bring them together both coherently and incoherently to look for excesses that are narrow in LST, frequency, or both. This presented some novel challenges which necessitated a new notebook specially designed for post-LST-stacked data. We had to overcome three challenges:

- First, the autocorrelations—which we use for modeling the noise—showed some temporal and (on the edges) spectral structure that risked creating artificial structure in our SNRs. We smoothed them with an iterative approach to avoid spurious structure, blacklisting certain LSTs and frequencies where artifacts from flagging and inpainting/expainting might persist.

- Second, we had to avoid introducing spectral structure into the SNRs by treating N_{samples} too literally. For the purpose of calculating noise, we replaced N_{samples} by its median above or below FM (much like what we do during power spectrum estimation currently). This produces a distribution of SNRs which is not quite perfect when we insert pure white noise, but it is close enough to not significantly bias our identification of outliers.

- Finally, we modified our filtering and leverage corrections to allow for both a main-beam fringe-rate top hat filter as well as a FR=0 notch filter. This was not done in Rounds 4 and 5 out of expediency and

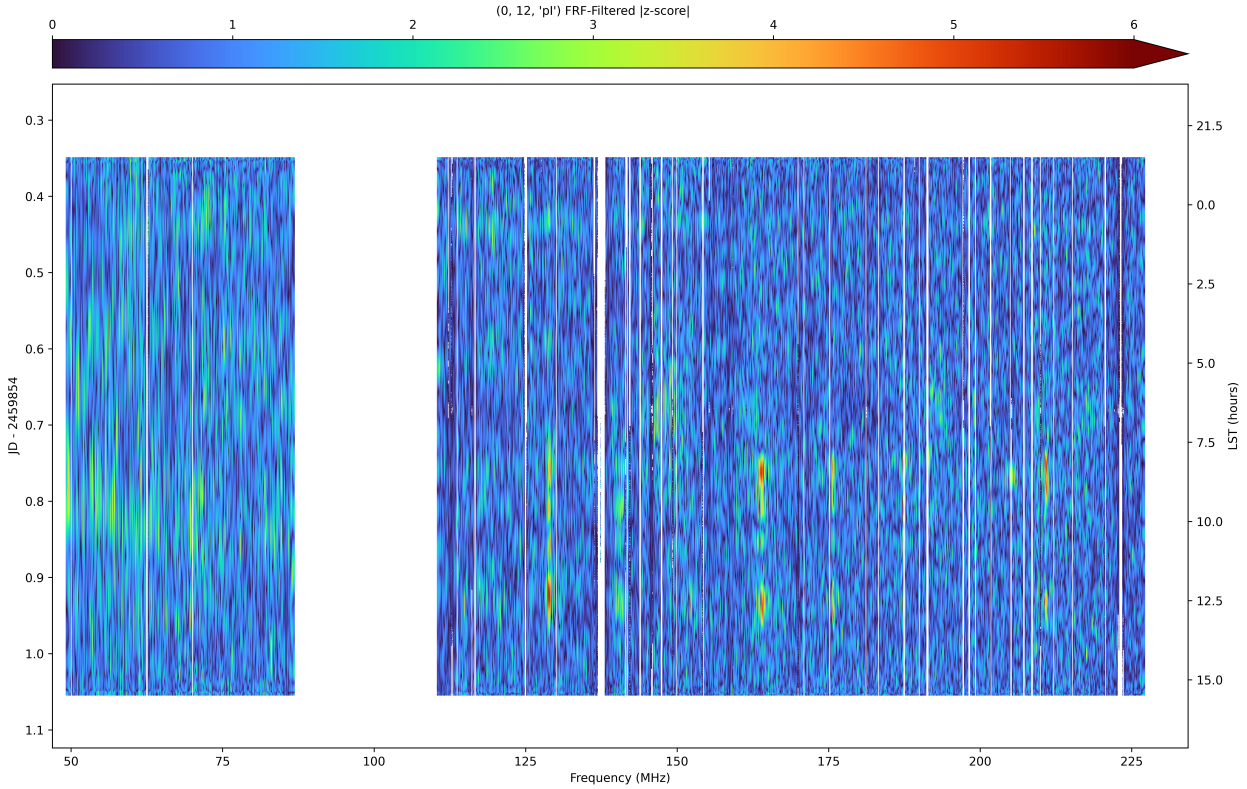

Figure 18: A handful of baselines (mostly those with short projected East-West distances) exhibit very clear spectral structure after delay- and fringe-rate filtering and forming properly normalized z-scores. The origin of this effect remains unknown, though it's likely related to the one we saw in Figure 10.

a desire to reduce the complexity of the notebook. However, this may have been a strategic mistake, since the clearest evidence for residual fringing RFI is on baselines with short projected East-West distances that were excluded from the Rounds 4 and 5 metrics due to the overlap of the main beam with fringe-rate 0.

However, upon bringing together many such SNR metrics in a Round 6 notebook, we do not find clear evidence that the effect is universal across baselines, and we conclude that the methods used in Rounds 4 and 5 are still appropriate. The presence of clear residual RFI in a few baselines with short East-West projected distances, such as that shown in Figure 18, are not made any clearer by averaging together many baselines incoherently or coherently on our list of sources of concern.

Given the lack of success in flagging this structure and not dramatically overflagging, *the flags written to disk during Round 6 are not currently used in any subsequent analysis*. The question of what to do about this structure and whether to flag particular baselines or LSTs or channels remains open. The Round 6 notebook includes Figure 18 as well as analogous plots for all baselines with any z-scores above 4.0, which should serve as a useful basis for triaging and investigating baselines with residual spectral structure.

5.2 Re-Inpainting and Flux Scale Correction

If we had used any of the Round 6 flag waterfall, we would have needed to re-inpaint every baseline, directly analogous to what we did in subsubsection 3.3.6 with the re-inpainting notebook. We have built an analogous notebook for re-inpaiting post-LST-stacking. The parameters are the same and the basic idea—to use the data as its own scaffold for feathered weights—is the same. Since we are not adding any flags, the only re-inpainting that is done is of 100% inpainted data, since in theory this should do a slightly better job of enforcing spectral smoothness. In practice, this is probably unnecessary.

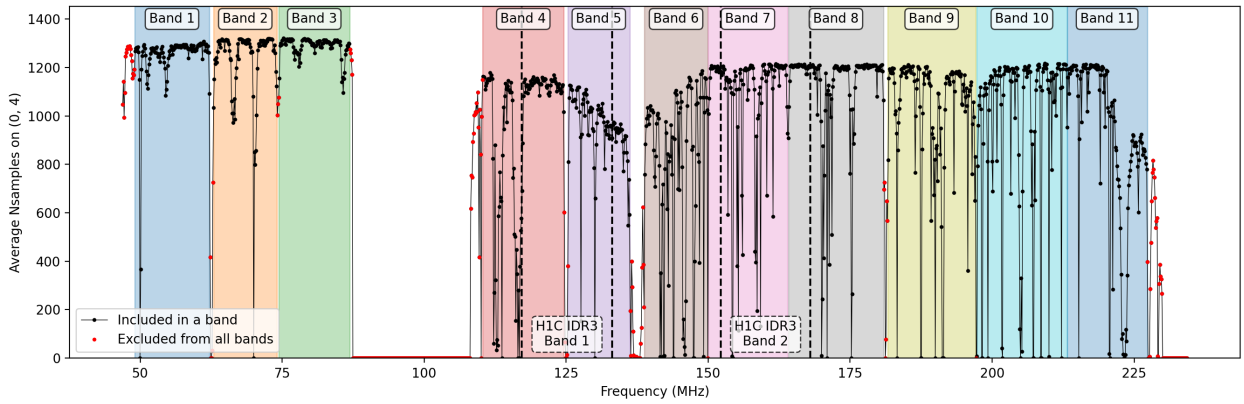

Figure 19: Given the increased flagging as a result of Rounds 3, 4, and 5 RFI excision, it seemed reasonable to reassess the power spectrum bands for this analysis. We used a similar process as in HERA Collaboration et al. (2025), attempting to put large flagging gaps at band edges. The increased flagging since then means fewer "clean" bands exist to try to isolate, leading to a choice of 11 slightly wider bands. We also walked away from the edges of both below FM and above FM, since they show the most artifacts from inpainting, expainting, and filtering. We may want to reconsider these bands in the future, though we should be careful to do so without reference to the final power spectrum.

However, this notebook has one important additional step: a final flux scale correction. In Hewitt (in prep.), it is shown that the flux scale we use in absolute amplitude calibration (based on the SSM model) produces point source fluxes that diverge from catalog fluxes at high frequency, especially above 200 MHz. This can be largely corrected by assuming that the sky and beam models are correct, but that the receiver temperature is low. Because we calibrate off the autocorrelations and because the receiver temperature is a more significant fraction of the total system temperature at high frequency, the correction is largest at high frequency.

To actually apply the correction, we use the ratio of the corrected SSM autocorrelations to the original SSM autocorrelations, cubic spline upsampled in time and frequency (as in the `file_calibration` notebook). To avoid introducing temporal or spectral structure, we fit with a 2D DPSS model with a time and frequency scale that matches `smooth_cal` before applying it to every visibility.

5.3 Single-Baseline Post-Processing and Power Spectra

The `single_baseline_postprocessing_and_pspec` notebook is largely unchanged from HERA Collaboration et al. (2025), with one notable exception. We use the edges of the first and last power spectrum bands above and below FM to set the ranges over which we filter for Round 6 and over which we re-inpaint. Therefore, while we have postponed the choice of power spectrum bands as long as possible, we now need to pick them. The bands are shown in Figure 19 and specified in Table 1.

Two other caveats:

- The products coming out of the single baseline power spectrum notebook do not currently include window functions. The near-term plan is to incorporate the window function calculation into the pipeline, either as a separate step or as part of the single baseline power spectrum notebook. These window functions are only calculated for the time-averaged power spectra, not the time-dependent ones, since calculating and storing time-dependent window functions and/or covariance matrices becomes very expensive.

- Right now, inpainted data are treated as contributing 0 samples, even though they add a low-noise-at-high-delay visibility to the average. As a result, error bars are overestimates, and our $P(k)/P_N(k)$ histograms at high delay show a clear deficit of noise relative to the expectation. This is better than underestimating the noise, but we do not currently have a more principled approach to the problem. Most of the inpainting is in a relatively small fraction of the channels, so being able to give them zero

Band	Channel Range	N_{chans}	Freq. Range (MHz)	$\Delta\nu$ (MHz)	z Range	Center z	Δz
1	18 – 126	108	49.1 – 62.2	13.2	21.82 – 28.0	24.9	6.1
2	131 – 223	92	62.9 – 74.1	11.2	18.17 – 21.6	19.9	3.4
3	226 – 328	102	74.4 – 86.9	12.5	15.35 – 18.1	16.7	2.7
4	520 – 637	117	110.3 – 124.6	14.3	10.40 – 11.9	11.1	1.5
5	643 – 732	89	125.4 – 136.2	10.9	9.43 – 10.3	9.9	0.9
6	753 – 844	91	138.8 – 149.9	11.1	8.48 – 9.2	8.9	0.8
7	845 – 960	115	150.0 – 164.0	14.0	7.66 – 8.5	8.1	0.8
8	960 – 1098	138	164.0 – 180.9	16.8	6.85 – 7.7	7.3	0.8
9	1104 – 1231	127	181.6 – 197.1	15.5	6.21 – 6.8	6.5	0.6
10	1232 – 1363	131	197.3 – 213.2	16.0	5.66 – 6.2	5.9	0.5
11	1363 – 1478	115	213.2 – 227.3	14.0	5.25 – 5.7	5.5	0.4

Table 1: The 11 frequency bands used for power-spectrum analysis, with their channel ranges, frequency ranges, bandwidths, and corresponding redshift ranges.

weight in the power spectrum estimator without introducing spectral structure would substantially alleviate the problem.

6 Summary of Data Products

Here we list as a quick reference all the data products available as part of this IDR, in the order they are produced by the two analysis pipelines—`h6c_analysis.toml` followed by `post_inpaint_analysis.toml`—and the LST-stacking pipeline—`single_bl_lst_stack.toml`.

6.1 Per-Night Data Products

Unless otherwise noted, each product lives on lustre at `/lustre/aoc/projects/hera/h6c-analysis/IDR3/<JD>/`. Products tagged "cold storage only" live at `/home/herastore02-4/analysis_cold_storage/<JD>_analysis/`; when the tag also names a `.tar.gz`, the product exists only inside that archive. All notebooks are linked here. However, many notebooks are run for every file (or after redundant averaging and corner turning, for every baseline). That full set of notebooks is generally zipped up and moved to cold storage, while a single representative copy (the middle file of each night, or the 4-unit EW baseline) is preserved on the NRAO website linked above.

- `zen.<JD>.?????.sum.calibration_notebook.html` (full set: cold storage only, in `<JD>_calibration_notebooks.tar.gz`): This is the per-file calibration notebook, which includes plots and tables detailing antenna flagging, preliminary RFI excision, and redundant and absolute calibration results. It produces the following:
  - `zen.<JD>.?????.sum.ant_metrics.hdf5` (cold storage only, in `<JD>_ant_metrics.tar.gz`): Summary of information about flagging produced by `hera_qm.ant_metrics`, which looks for dead, low-correlation, and cross-polarized antennas.
  - `zen.<JD>.?????.sum.ant_class.csv`: Summary of statistics used for antenna classification as good, bad, or suspect.
  - `zen.<JD>.?????.sum.omni.calfits` (cold storage only, in `<JD>_omni_calfits.tar.gz`): Gain solutions that include both redundant-baseline calibration and absolute calibration, as well as preliminary RFI flagging.
- `antenna_classification_summary_<JD>.html`: This per-night notebook summarizes the reasons antennas were flagged over a whole night.
- `full_day_rfi_<JD>.html`: This per-night notebook performs the first full-day RFI flagging pass by looking for outliers in 2D-DPSS-filtered averaged autocorrelations. It produces the following:

  - zen.<JD>.?????.sum.flag_waterfall.h5 (cold storage only, in <JD>_flag_waterfalls.tar.gz): Per-file RFI flag waterfall from the first-pass RFI search.

- full_day_antenna_flagging_<JD>.html: This per-night notebook harmonizes and expands antenna flagging across the whole day, adding flags to often-but-not-always flagged antennas coming out of per-file processing. It produces the following:
  - zen.<JD>.?????.sum.antenna_flags.h5: Per-file antenna-polarization flags after full-day harmonization (each 2-integration ant-pol waterfall is either fully flagged or fully unflagged).

- calibration_smoothing_<JD>.html: This per-night notebook performs calibration smoothing across the full day in time and frequency, also looking for and reporting phase flips. It produces the following:
  - zen.<JD>.?????.sum.smooth.calfits (cold storage only, in <JD>_smooth_calfits.tar.gz): Smoothed calibration solutions including the antenna and round-2 RFI flags (updated in place from round-1 RFI flags), with gains smoothed in time and frequency.

- zen.<JD>.?????.sum.delay_filtered_average_zscore_notebook.html (full set: cold storage only, in <JD>_delay_filtered_average_zscore_notebooks.tar.gz): This per-file notebook computes a delay-filtered, redundantly-averaged z-score metric used as input to the second round of full-day RFI flagging. It produces the following:
  - zen.<JD>.?????.sum.red_avg_zscore.h5 (cold storage only, in <JD>_red_avg_zscores.tar.gz): Per-file delay-filtered, redundantly-averaged z-score waterfall used to identify additional RFI.

- full_day_rfi_round_2_<JD>.html: This per-night notebook performs a second full-day RFI flagging pass using the delay-filtered z-scores. It updates the zen.<JD>.?????.sum.smooth.calfits files in place. It produces the following:
  - zen.<JD>.?????.sum.flag_waterfall_round_2.h5: Per-file RFI flag waterfall after the second-pass search.
  - <JD>_aposteriori_flags.yaml: A YAML list of JD ranges, frequency ranges, and antennas that are 100% flagged over the day, used downstream to keep redundantly-averaged data products consistent.

- zen.<JD>.?????.sum.postprocessing_notebook.html (full set: cold storage only, in <JD>_postprocessing_notebooks.tar.gz): This per-file notebook applies the smoothed calibration solutions and round-2 flags, redundantly averages, and visualizes the results. It produces the following:
  - zen.<JD>.?????.sum.smooth_calibrated.bl_select.uvh5: A baseline-selected, decimated subset of the 30 most redundant baseline groups, with smooth-calibrated applied, but *without* redundant averaging. These are only produced once every 30 files and used to study non-redundancy.
  - zen.<JD>.?????.sum.smooth_calibrated.avg_abs_all.uvh5 (cold storage only, in <JD>_avg_abs.tar.gz): Smooth-calibrated, redundantly-averaged visibility amplitudes incoherently averaged over all baselines.
  - zen.<JD>.?????.sum.smooth_calibrated.avg_abs_auto.uvh5 (cold storage only, in <JD>_avg_abs.tar.gz): Same as above but restricted to autocorrelations.
  - zen.<JD>.?????.sum.smooth_calibrated.avg_abs_cross.uvh5 (cold storage only, in <JD>_avg_abs.tar.gz): Same as above but restricted to cross-correlations.
  - zen.<JD>.?????.sum.reds_used.p: Pickle of the list of lists of unflagged redundantly-averaged baseline groups actually used for this file.

  Note that the actual redundantly averaged files produced, zen.<JD>.?????.sum.smooth_calibrated.red_avg.uvh5 are not kept, since they are superseded by (and reconstructible from) corner-turned, (re-)inpainted data.

- single_baseline_files/corner_turn_map.yaml: Per-night YAML mapping (built by the build_corner_turn_map.py script) from per-file red-avg files to single-unique-baseline files for the corner turn. Note that the outputs of the corner-turn, single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.uvh5, are not kept since they are also superseded by (re-)inpainted data.

- single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.single_baseline_2D_filtered_SNRs.html (full set: cold storage only, in <JD>_single_baseline_2D_filtered_SNRs_notebooks.tar.gz): This per-baseline notebook 2D-DPSS filters the most redundant baselines to produce SNR waterfalls used for the third RFI round. It produces the following:

  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.2Dfilt_SNR.uvh5 (cold storage only, in <JD>_2Dfilt_SNRs.tar.gz): Per-baseline 2D-DPSS-filtered SNR waterfalls for the most redundant baselines.

- full_day_rfi_round_3_<JD>.html: This per-night notebook performs a third full-day RFI flagging pass by combining the per-baseline 2D-DPSS-filtered SNRs across all highly-redundant baselines. It produces the following:

  - single_baseline_files/zen.<JD>.flag_waterfall_round_3.h5: Per-night RFI flag waterfall after the third-pass search.

- single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.single_baseline_2D_informed_inpaint.html (full set: cold storage only, in <JD>_single_baseline_2D_informed_inpaint_notebooks.tar.gz): This per-baseline notebook performs 2D-DPSS-informed inpainting in time and frequency on each redundantly-averaged single-baseline file using the round-3 flags. It produces the following:

  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.inpainted.uvh5 (cold storage only): Per-baseline redundantly-averaged visibilities with flagged channels inpainted.
  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.where_inpainted.h5 (cold storage only): Boolean mask indicating which time–frequency cells were inpainted.

- full_day_systematics_inspect_<JD>.html: This per-night notebook examines full-day waterfalls in frequency/time, delay/time, and delay/fringe-rate space before and after delay and fringe-rate filtering, as well as coherent and incoherent averaging and after forming pseudo-Stokes I and Q. It produces no data products. This notebook is largely superseded by full_day_systematics_inspect_post_reinpaint_<JD>.html (see below).

- single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.single_baseline_pI_SNR.html (full set: cold storage only, in <JD>_single_baseline_pI_SNR_notebooks.tar.gz): This per-baseline notebook computes pseudo-Stokes I delay-filtered SNRs (without fringe-rate filtering) on the inpainted single-baseline files with the highest redundancy. It produces the following:

  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.inpainted.pI_DLYFILT_SNR.uvh5 (cold storage only, in <JD>_pI_DLYFILT_SNRs.tar.gz): Per-baseline delay-filtered pseudo-Stokes I SNR waterfalls of the highest redundancy baselines.

- full_day_rfi_round_4_<JD>.html: This per-night notebook performs a fourth full-day RFI flagging pass by coherently rephasing the per-baseline delay-filtered pseudo-Stokes I SNRs to a set of known bright sources and looking for outliers. It produces the following:

  - single_baseline_files/zen.<JD>.<SOURCE>.flag_waterfall_round_4.h5: Per-night RFI flag waterfall for each rephasing source <SOURCE>.

- single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.single_baseline_pI_FRF_SNR.html (full set: cold storage only, in <JD>_single_baseline_pI_FRF_SNR_notebooks.tar.gz): This per-baseline notebook re-computes pseudo-Stokes I SNRs after main-beam fringe-rate filtering the most redundant baselines, using the round-4 flags as a prior. It produces the following:
  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.inpainted.pI_FRF_SNR.uvh5 (cold storage only, in <JD>_pI_FRF_SNRs.tar.gz): Per-baseline fringe-rate-filtered pseudo-Stokes I SNR waterfall.
- full_day_rfi_round_5_<JD>.html: This per-night notebook performs a fifth full-day RFI flagging pass on the fringe-rate-filtered pseudo-Stokes I SNRs, both rephased to known bright sources and incoherently stacked across baselines. It produces the following:
  - single_baseline_files/zen.<JD>.<SOURCE>.flag_waterfall_round_5.h5: Per-night RFI flag waterfall for each rephasing source <SOURCE>, plus an additional waterfall with <SOURCE>=incoherent for the incoherent baseline stack.
- single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.single_baseline_scaffolded_and_feathered_inpainter.html (full set: cold storage only, in <JD>_single_baseline_reinpaint_notebooks.tar.gz): This per-baseline notebook performs scaffolded-and-feathered 1D DPSS re-inpainting on each single-baseline file using the updated round-5 flags, with the previously inpainted data acting as the scaffold. It produces the following:
  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.reinpainted.uvh5: Per-baseline re-inpainted visibilities consistent with the round-5 flags.
  - single_baseline_files/zen.<JD>.baseline.<a1>_<a2>.sum.smooth_calibrated.red_avg.where_reinpainted.h5: Boolean mask indicating which time–frequency cells were re-inpainted in this round.

  These are the final, per-night data products used in LST-stacking.
- full_day_systematics_inspect_post_reinpaint_<JD>.html: This per-night notebook re-runs the full-day systematics inspection (delay filtering, frequency–time inpainting, cross-talk filtering, and main-beam fringe-rate filtering) on the re-inpainted data. It produces no data products.

6.2 LST-Stacked Data Products

All of the following data products are available on lustre at /lustre/aoc/projects/hera/h6c-analysis/IDR3/lststack-outputs-131-nights/. The LST-stacking pipeline is specified by single_bl_lst_stack.toml and operates on the per-night, per-baseline re-inpainted visibilities described above. It stacks 131 nights of data in LST, performing outlier rejection and re-inpainting the stacked data. An optional FR=0 filter is also applied per-night before stacking, producing an additional set of output files. All notebooks are linked from the H6C IDR3 page.

- baseline_map.yaml: A YAML file (built by the BUILD_BASELINE_MAP task) mapping each unique baseline to its physical length, average redundancy, whether it is in the preliminary set used for LST-calibration, and which night anchors the LST-calibration for that baseline.
- baseline.<a1>_<a2>.preliminary.lst_stack_and_reinpaint.html: This per-baseline notebook performs a preliminary LST-stack (without LST-calibration) on the 30 most redundant baselines used to derive LST-calibration solutions. It produces the following:
  - single_bl_reinpaint_1000ns/zen.LST.baseline.<a1>_<a2>.preliminary.sum.uvh5: Per-baseline preliminary LST-stacked visibilities (without LST-calibration applied).
  - single_bl_reinpaint_1000ns/zen.LST.baseline.<a1>_<a2>.preliminary.sum.FR0filt.uvh5: Same as above but with a per-night FR=0 filter applied before stacking. These are not currently used.

- zen.<JD>.single_night_lstcal.html: This per-night notebook derives LST-calibration solutions (amplitude, tip-tilt phase, and cross-polarization phase) by comparing the preliminary LST-stacked data to the per-night data. It produces the following:
  - single_bl_reinpaint_1000ns/zen.<JD>.lstcal.hdf5: Per-night HDF5 file containing the LST-cal gain solutions, smoothed in frequency and time, to be applied during the final LST-stack.
- baseline.<a1>_<a2>.lst_stack_and_reinpaint.html: This per-baseline notebook performs the final LST-stack with LST-calibration applied, stacking all nights at each LST bin, performing modified z-score outlier rejection of poorly inpainted data and re-inpainting. It produces the following:
  - single_bl_reinpaint_1000ns/zen.LST.baseline.<a1>_<a2>.sum.uvh5: Per-baseline final LST-stacked, LST-calibrated, and re-inpainted visibilities.
  - single_bl_reinpaint_1000ns/zen.LST.baseline.<a1>_<a2>.sum.FR0filt.uvh5: Same as above but with a per-night FR=0 filter applied before stacking.

6.3 Post-LST-Stacked Data Products

All of the following data products are available on lustre at /lustre/aoc/projects/hera/h6c-analysis/IDR3/pspec-outputs-131-nights/. The post-LST-stack power-spectrum pipeline is specified by h6c_pspec_11band.toml. It performs metrics for a sixth round of RFI flagging on the LST-stacked data—which is not currently used—re-inpaints the result, and computes power spectra in 11 frequency bands spanning 49.0–227.3 MHz (with the band edges enumerated in Table 1) over the LST range $[22.5, 15.5]$ hours. The pipeline operates on the per-baseline LST-stacked visibilities (zen.LST.baseline.<a1>_<a2>.sum.uvh5). All notebooks are linked from the H6C IDR3 page.

- zen.LST.baseline.<a1>_<a2>.sum.single_loststack_baseline_pI_FRF_SNR.html: This per-baseline notebook forms pseudo-Stokes I SNR waterfalls and applies delay filtering, FR=0 notching, and per-channel main-beam fringe-rate filtering. It produces the following:
  - zen.LST.baseline.<a1>_<a2>.sum.pI_FRF_SNR.uvh5: Per-baseline delay- and fringe-rate-filtered pseudo-Stokes I SNR waterfall, written only for baselines meeting the minimum sampling requirement. *Note: these are not currently used in the pipeline.*
- full_loststack_rfi_round_6.html: This single per-LST-run notebook aggregates the per-baseline filtered SNRs, coherently rephases them to a set of bright sources, also computes an incoherent stack across all baselines, and applies iterative median/mean z-score flagging together with time-convolution thresholding and watershed clustering. It produces the following:
  - zen.LST.<SOURCE>.flag_waterfall_round_6.h5: LST-stacked RFI flag waterfall for each rephasing source <SOURCE> $\in$ {PSR_J0628-28, PSR_J0742-2822, PSR_J0835-4510, Fornax_A, Pictor_A, Crab, Sag_A, Cen_A, Cyg_A, Cass_A}, plus an additional waterfall with <SOURCE>=incoherent for the incoherent baseline stack.
- zen.LST.baseline.<a1>_<a2>.sum.single_loststack_baseline_scaffolded_and_feathered_inpainter.html: This per-baseline notebook performs a final pass of scaffolded-and-feathered 1D DPSS re-inpainting along frequency on each LST-stacked single-baseline file. It also applies a 2D-DPSS-smoothed SSM gleam-scale absolute-calibration correction. Only nsamples==0 visibilities are currently re-inpainted, as no flags from round 6 are applied. It produces the following:
  - zen.LST.baseline.<a1>_<a2>.sum.reinpainted.uvh5: Per-baseline LST-stacked visibilities with all nsamples==0 pixels re-inpainted.
- zen.LST.baseline.<a1>_<a2>.sum.single_baseline_postprocessing_and_pspec.html: This per-baseline notebook applies final pixel-, channel-, and integration-level flag cuts, masks the FM band, splits the data into four time interleaves, performs additional fringe-rate/crosstalk filtering, converts to pseudo-Stokes, and computes power spectra in each of the 11 bands using a Blackman-Harris taper, with fringe-rate-loss bookkeeping. It produces the following:

  - `zen.LST.baseline.<a1>_<a2>.sum.reinpainted.pspec.h5`: Per-baseline, per-band power spectra at the (coherently averaged) ∼300 s time resolution, with the four interleave-pair pspecs incoherently averaged at each time.
  - `zen.LST.baseline.<a1>_<a2>.sum.reinpainted.tavg.pspec.h5`: Same as above, additionally incoherently averaged in time across the full LST range to give a single spectrum per band per baseline.

- `baselines_merged.sum.reinpainted.pspec.h5` and `baselines_merged.sum.reinpainted.tavg.pspec.h5`: Baseline-averaged power spectra produced by the `MERGE_SINGLE_BASELINE_FILES` task, which runs `pspec fast-merge-baselines` to combine the per-baseline pspec files above, grouped by Stokes polarization, with fringe-rate-loss extras preserved. This task has no associated notebook.

In parallel, the same pipeline was run on the per-night FR=0-filtered LST-stacked visibilities (`zen.LST.baseline.<a1>_<a2>.sum.FR0filt.uvh5`, described in the previous subsection), using a slightly modified `toml`. The equivalent results files can all be found in `/lustre/aoc/projects/hera/h6c-analysis/IDR3/pspec-outputs-131-nights-FR0filt/` and are generally labeled `.sum.FR0filt.` instead of `.sum.`. The notebooks for this run live alongside the version without the FR=0 notch filter during LST-stacking, and can be found in single_lststack_baseline_pI_FRF_SNR, full_lststack_rfi_round_6, single_lststack_baseline_scaffolded_and_feathered_inpainter, and single_baseline_postprocessing_and_pspec.

7 Open Questions and Future Directions

Here are some areas for future inquiry and improvement that we ought to address as we move toward publishing a limit and toward analyzing the rest of HERA Phase II:

- Tracking down Round 6 anomalies: the notebook designed to find and surface high SNR points in time and frequency waterfalls after delay- and fringe-rate-filtering reveals a number of high SNR streaks and blobs that ought to be tracked down. Some of these look like the same fringing RFI we designed Rounds 4 and 5 to detect and flag on a per-night basis, especially on baselines where the main beam fringe rates overlap with FR=0 (and were thus excluded from the creation of Round 4 and 5 flags—though this could change with a re-run). Right now, the notebook's flags point toward dramatic overflagging and are not trusted and not used. Whether we want to be flagging or filtering by shape or something else is a matter that requires investigation.

- Investigating and perhaps localizing the fringing RFI: The kind of compact-in-frequency structure we currently best identify in delay- and fringe-rate-filtered, incoherently averaged visibilities (see Figure 10) has generally not been attributed to a particular source or even physical mechanism. And frankly, given the long timescales apparently involved, it's quite confusing! While we seem to be doing a pretty good job of flagging and removing it with our Rounds 4 and 5 RFI search, more work is warranted here to help detect it when it shows up at a low level, as well as to help understand to what extent it affects all baselines and/or antennas vs. just a subset. Speculatively, one wonders whether there's a relationship between it and the aliased RFI we saw (and largely flagged) in Figure 2 of Dillon et al. (2024).

- Additional phased searches for RFI: we've had a lot of success phasing to particular celestial sources to discover spectral structure due to scintillation, revealing phenomena that could only be seen sketchily in incoherent averages over baseline. It follows then that we might pursue a similar approach to trying to discover sources of manmade RFI, both terrestrial and from satellites. We should be exploring coherently averaged waterfalls phased to many points along the horizon, especially directions of known transmitters. We should also be using known satellite positions to perform phased searches for RFI from ORBCOMM and Starlink.

- Modeling and removing pulsars: our current Rounds 4 and 5 flagging finds substantial spectral structure attributable to PSR J0628-28 and flags it—often leading to nearly an hour of LST around the pulsar's transit being flagged. A similar search for pulsar scintillation due to PSR J0742-2822 does not find nearly as much, so the LST is not nearly as flagged. However, we do see evidence for $pI \to pQ$

leakage at LSTs and delays clearly attributable to PSR J0742-2822's RM. One could simply flag these delays in each power spectrum band, which may be the simplest solution for mitigating the effect. However, recent work has shown that both pulsars can be modeled and removed, fitting their RM and scintillation effects with relatively few degrees of freedom (and thus very little signal loss). This may be a bit expensive, since it requires going back to the per-day, post-inpainting pipeline (see Figure 2) and adding a couple of new steps, but it is also likely the right way to handle the issue going forward. We should also look for high RM pulsars at other LSTs, utilizing pQ which shows substantially more structure than pI but which is probably leaking into pI at some low level.

- Apparent scintillation from other sources: a further step would be to try to null out flux in the direction of other bright sources, especially Fornax A, which shows evidence of a different sort of scintillation (perhaps interplanetary, perhaps ionospheric, but certainly on a faster timescale than the ISM scintillation we see so clearly on PSR J0628-28). This will be more difficult than the pulsars, since Fornax A is marginally resolved, but perhaps there's still an opportunity here to reduce the flagging in Round 4 by subtracting a few bright A-team sources.

- Improving FR=0 filtering in the LST-stacker: right now, the LST-stacker filters each individual day to remove the FR=0 mode. The idea is that inhomogenous flagging patterns in time could spread intrinsically low fringe-rate structure to higher fringe-rates, by analogy to the way LST-stacking with inhomogenous sampling and no inpainting can create spectral structure (albeit at a far higher level). The question is, should we be using these data products? Some baselines show spurious time and frequency structure, likely due to the way re-inpainting and expainting interact with the filter. However, we have preliminary evidence that they are broadly helpful in lowering the leakage and improve the power spectra across delays. There's also a question of whether 0.01is the right width for this filter, or if it should be bumped up, trading signal loss on some baselines for systematics suppression. This should be investigated and improved, ideally before the next big re-run of the LST stacker.

- Integration of window function calculations into the pipeline: right now, approximate window functions (i.e. those that follow the Gorce et al. (2023) prescription but do not fully account for inpainting as Chen et al. (2024) did prior to scaffolded and feathered inpainting) were calculated externally to the single-baseline postprocessing and power spectrum notebook for H6C IDR2 and for HERA Collaboration et al. (2025). Work to better integrate them so that window functions are always computed and added automatically to power spectrum data products is underway, but hasn't been completed yet.

- Delay filtering: we are generally limited at low k by foreground leakage. To what extent this is an effect of tapering that might be mitigated by wide-band DPSS filtering before forming narrow-band power spectra (Ewall-Wice et al., 2021) remains to be investigated. It's possible that mutual coupling dominates at these delays anyway, so no improvement is available to be had. One might also wonder whether there's any benefit to delay filtering before LST-stacking.

- Cleanup of power spectrum summary notebooks: the two notebooks responsible for summarizing the output of the single baseline power spectra to produce waterfall statistics and per-baseline and per-band statistics, wedges, and final $\Delta^2(k)$ have a lot of hard-coded choices appropriate and expedient for H6C IDR2. While designed to be run interactively and thus they don't necessarily need to be run automatically as part of the pipeline (though perhaps they should), they do need some work and better testing to make sure the final results are correct.

- Power spectrum estimators with non-uniform weighting: right now, we treat inpainted data as if it were 0 samples for the purpose of writing down the noise covariance. Of course, that's not true and it leads us to overestimate the noise, especially on highly inpainted bands. We need a power spectrum estimator that lets us down-weight heavily inpainted channels. While we cannot move completely away from inpainting without losing a ton of sensitivity (since many channels at a given LST are only flagged for a small fraction of nights), we could substantially reduce our reliance on it, given how much low-level RFI we've found and thus how difficult it is to pick truly clean bands (see Figure 19).

- Better integrating the pre-and post-inpainting pipelines: the split of the per-night pipeline into two parts (see Figure 1 and Figure 2) is purely a historical artifact of the way this analysis was conducted

and the gradual discovery of the need for better and better RFI excision. These two can be combined and should be, eventually, perhaps also condensing some RFI excision steps. Whether that should be done for an H6C reanalysis or only for the rest of HERA Phase II is a strategic question that remains open.

- Rebuilding calibration and inpainting to use LST-stacked data: right now, the most expensive part of the pipeline is the first inpaint step, where we build successively higher delay 2D DPSS models to scaffold the 1D inpaint. That was appropriate when we didn't have deep LST-stacked data. For future analyses of these same baselines as in this IDR, we should avoid this step and rely instead on LST-stacked data as a scaffold, perhaps with low-delay "un-calibration" to make it better match the data we're inpainting. We can take this further though: we should eliminate the `abscal` model in favor of this LST-stacked data set. We may even want to eliminate `redcal` and calibrate directly against it. This needs a thorough investigation, but it may allow us to dramatically simplify the pipeline and perhaps reduce our reliance on `smooth_cal`.

- Outriggers: thus far, no attempt has been made to calibrate the outrigger antennas; they are simply flagged during the initial calibration. As we rely increasingly on modeling and subtracting point sources like pulsars, we may benefit from including outriggers. Ancillary science may also benefit. This risks increasing the volume of redundantly averaged data by a factor of up to 4 (and thus will make expensive steps after the redundant average a lot more expensive, especially inpainting), so we will need to carefully weigh the costs and benefits of expanding our analysis to include outriggers.

References

Bull, P. (2024). "HERA Memo #129: A quick primer on the HERA DPSS fringe-rate filter implementation". reionization.org/memos.

Chen, Kai-Feng, Michael J. Wilensky, Adrian Liu, et al. (2024). *Impacts and Statistical Mitigation of Missing Data on the 21cm Power Spectrum: A Case Study with the Hydrogen Epoch of Reionization Array*. arXiv: 2411.10529 [astro-ph.CO]. URL: https://arxiv.org/abs/2411.10529.

Dillon, J. S. and T. A. Cox (2024). "HERA Memo #133: Improvements to Flagging of Low-Level RFI". reionization.org/memos.

Dillon, J. S. and S. Murray (2023a). "HERA Memo #124: H6C Internal Data Release 2.1". reionization.org/memos.

— (2023b). "HERA Memo #125: H6C Internal Data Release 2.2". reionization.org/memos.

Dillon, J. S., S. Murray, T. A. Cox, et al. (2024). "HERA Memo #131: H6C Internal Data Release 2.3". reionization.org/memos.

Ewall-Wice, Aaron, Nicholas Kern, Joshua S. Dillon, et al. (Jan. 2021). "DAYENU: a simple filter of smooth foregrounds for intensity mapping power spectra". In: MNRAS 500.4, pp. 5195–5213. DOI: 10.1093/mnras/staa3293. arXiv: 2004.11397 [astro-ph.CO].

Gorce, Adélie, Samskruthi Ganjam, Adrian Liu, et al. (Jan. 2023). "Impact of instrument and data characteristics in the interferometric reconstruction of the 21cm power spectrum". In: *Monthly Notices of the Royal Astronomical Society* 520.1, 375–391. ISSN: 1365-2966. DOI: 10.1093/mnras/stad090. URL: http://dx.doi.org/10.1093/mnras/stad090.

HERA Collaboration, The, Zuhra Abdurashidova, Tyrone Adams, et al. (2025). "First Results from HERA Phase II". In: arXiv: 2511.21289 [astro-ph.CO]. URL: https://arxiv.org/abs/2511.21289.

Hewitt, J. N. (in prep.). "HERA Memo #???: A First Look at the HERA Flux Density Scale: Dirty-map Beam Cuts of Bright Sources".

Keating, Garrett K., Bryna Hazelton, Matthew Kolopanis, et al. (2024). "Memo: Phasing in pyuvdata". https://github.com/RadioAstronomySoftwareGroup/pyuvdata/blob/main/docs/references/phasing.pdf.

Murray, S. and J. S. Dillon (2023). "HERA Memo #122: H6C Season A Priori Flag Summary". reionization.org/memos.

Pascua, Robert, Zachary E. Martinot, Adrian Liu, et al. (2024). *A Generalized Method for Characterizing 21-cm Power Spectrum Signal Loss from Temporal Filtering of Drift-scanning Visibilities*. arXiv: 2410.01872 [astro-ph.CO]. URL: https://arxiv.org/abs/2410.01872.

Storer, Dara, Joshua S. Dillon, Daniel C. Jacobs, et al. (2022). "Automated Detection of Antenna Malfunctions in Large-N Interferometers: A Case Study With the Hydrogen Epoch of Reionization Array". In: *Radio Science* 57.1. DOI: 10.1029/2021rs007376. URL: https://doi.org/10.1029%2F2021rs007376.