HERA Memo 129
Received 27 March 2024

A quick primer on the HERA DPSS fringe-rate filter implementation

Phil Bull

March 27, 2024

Abstract

This note outlines how the HERA DPSS/least-squares fringe-rate filter is implemented. We discuss the motivation for using DPSS functions to implement a bandpass (tophat-like) filter in the fringe-rate domain; define the DPSS basis functions mathematically; and describe the procedure for fitting the functions to data in order to do the filtering. The filtering code itself is provided by the `hera_filters` package, and much of the complexity is related to choosing how many DPSS modes to fit, and what method to use to find the 'best fit' to any given set of input data.

Overview

We are trying to implement a tophat filter in fringe-rate space – In broad outline, the HERA DPSS fringe-rate filter (FRF) implementation is intended to apply a flat (tophat) bandpass filter, which leaves some fringe-rate modes in the data intact, and excises others completely. At the moment, there is no attempt to implement a 'shaped' or 'weighted' filter that would apply a primary beam weighting for instance – all we care about for now is keeping or removing each FR mode. When we talk about the 'mainlobe' or 'notch' filters, we are always talking about keeping (or removing) all FR modes in some range; the 'mainlobe' filter doesn't mean that we are applying weights determined by the shape of the beam for example, only that we are using the shape of the mainlobe to define a range of fringe rates for the tophat.

Why not do a tophat filter in fringe-rate directly? – If we were to implement a tophat filter directly in FR space, we would see ringing on transforming back to the time domain. Instead, we can implement something that is close to a tophat filter, in the sense that it mostly only affects the modes inside the range defined by the tophat function, but which also reduces ringing in the time domain (i.e. doesn't scatter power to large separations in time). The functions that have the property of being 'maximally concentrated' in both the time *and* FR domains are the DPSS (Discrete Prolate Spheroidal Sequence) functions.

Relationship between DPSS functions and the tophat filter – The DPSS functions are defined with reference to a finite band within which we want to maximally concentrate the power of the function (i.e. we want to minimise the leakage of power outside this region). The concentration ratio of the function X in the fringe-rate domain is defined as

$$\lambda = \int_{f_c-\Delta f}^{f_c+\Delta f} |X(f)|^2 df \Big/ \int_{f_{\min}}^{f_{\max}} |X(f)|^2 df, \tag{1}$$

where we have defined the tophat region as $f_c - \Delta f \leq f \leq f_c + \Delta f$. Maximising this quantity, we obtain the first DPSS mode *for this particular tophat function*. We can then construct an orthonormal basis of DPSS functions by finding the next function that maximises λ while also being orthogonal to the first function, and so on. This procedure gives us a set of basis functions that keep as much of their power within the tophat band in the FR domain as possible.

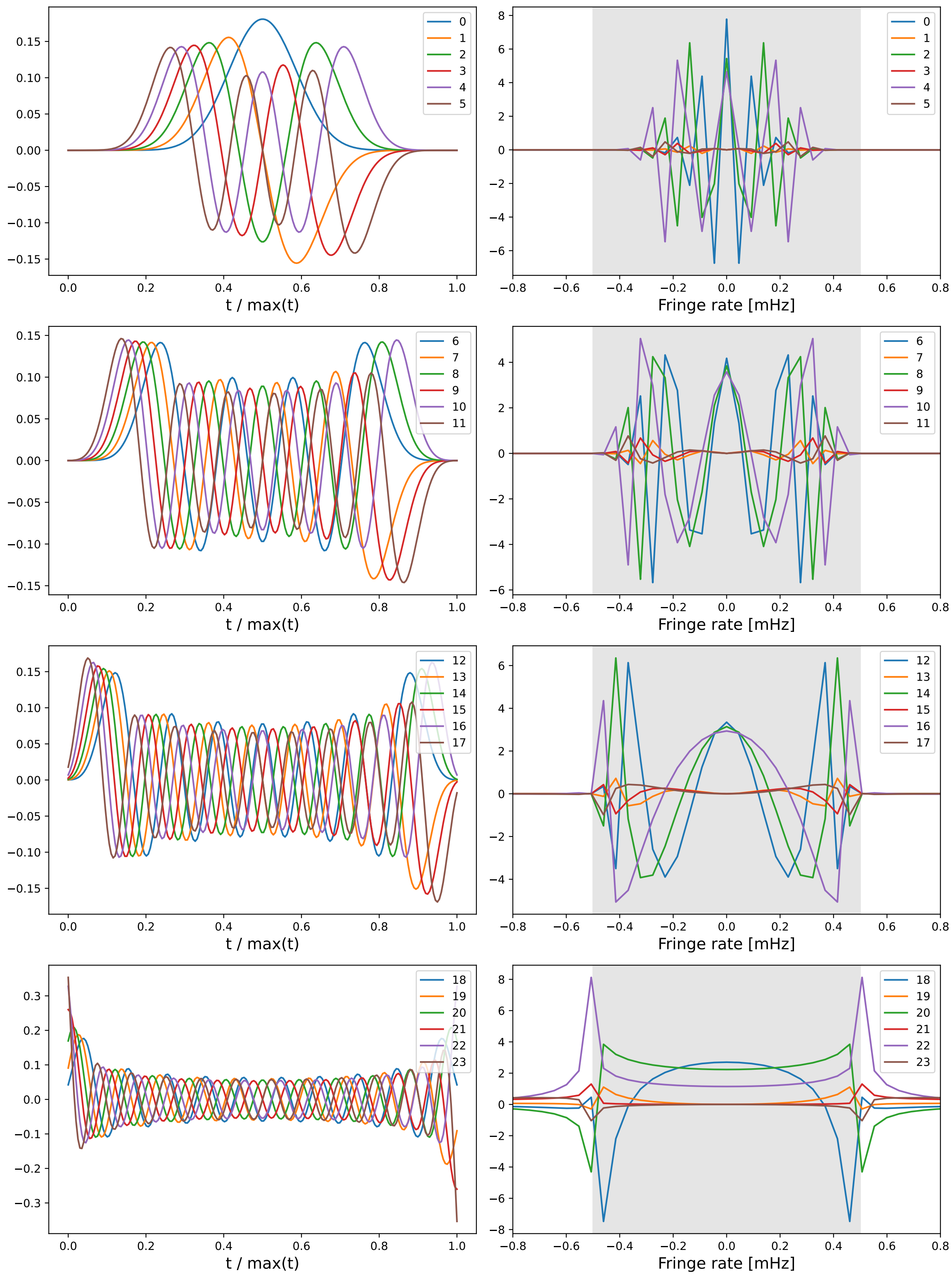

Figure 1: *(Left column):* Real part of the DPSS operator basis functions as a function of (normalised) time, for a given number of samples n_f, time sample duration, δt, and tophat window half-width Δf, and centre fringe rate f_c. We have chosen $f_c = 0$ for this example, so there is no imaginary part. *(Right panel):* Real part of the fringe-rate transform of the DPSS functions. The grey region shows the bounds of the tophat function.

Definition of the DPSS basis functions – We can define a matrix operator that has a DPSS basis function in each column,

$$A_{ij} = e^{2\pi i(t_i - t_c)f_c}\,\mathcal{D}_j(t_i;\;\Delta = n_f\,\delta t\,\Delta f), \tag{2}$$

where t_i is the time for sample i, t_c is the time at the centre of the observation period (of duration Δt), f_c is the fringe rate of the centre of the tophat filter, and $\mathcal{D}_j$ is the j'th DPSS function for a band with n_f regularly-spaced samples and half-bandwidth Δ, defined as the product of n_f, the time sample duration δt, and tophat half-width Δf. Note how the complex exponential is just a phase factor that depends on the tophat centre, f_c.

What do the DPSS basis functions look like? – Fig. 1 shows a set of DPSS modes (as defined in Eq. 2) for a given observing window, time sample duration, and tophat window centre and width. We have used $\Delta t = 6$ hours, $\delta t = 108.5$ sec ($n_f = 200$), $f_c = 0$ mHz, and $\Delta f = 0.5$ mHz for this example. The upper-left panel shows the lowest-order modes, which are concentrated near the centre of the observing time interval and taper nicely at the interval edges. Higher-order modes have more weight towards the edges, and the highest-order ones shown (bottom-left panel) are non-zero at the edges.

Projecting a data vector onto the DPSS modes – We can project a data vector onto our set of DPSS basis functions simply by doing (note Einstein summation convention)

$$c_j = A^\dagger_{ij} d_i, \tag{3}$$

where d_i is the element of a complex data vector for time sample i, c_j is the complex coefficient for DPSS mode j, and † denotes complex conjugation. Transforming a set of DPSS coefficients back to the time domain is achieved by doing

$$y_i = A_{ij} c_j. \tag{4}$$

We will typically only work with the leading order DPSS modes, up to some maximum order N, so projecting onto the DPSS modes and then back again will be lossy, i.e. $\vec{y} = AA^\dagger \vec{d} \neq \vec{d}$.

Linear least squares solution – If we only use $N < N_{\rm max}$ of the DPSS modes, and/or the data have missing values due to flagging, the DPSS basis as defined above is no longer complete and orthogonal with respect to our data, and so there is no unique way of arriving at a set of coefficients c_j. Instead, we must choose some method of defining a 'best-fit' set of coefficients that has sensible properties.

A common way of doing this is to find the least-squares solution $\hat{c}_j$ to the equation $A^\dagger_{ij} c_j = b_i$, where we would ideally define $b_i \equiv N^{-1}_{ik} d_k$ to incorporate an inverse noise covariance weighting of the data, but in reality will usually just use the data itself, $b_i \equiv d_i$. The generalised linear least-squares solution is

$$\hat{c}_j = \left(A^\dagger N^{-1} A\right)^{-1} A^\dagger N^{-1} \vec{d}, \tag{5}$$

which for unweighted data ($N = I$) reduces to

$$\hat{c}_j = \left(A^\dagger A\right)^{-1} A^\dagger \vec{d}. \tag{6}$$

The (unweighted) linear least-squares DPSS **filtered data** can then be written as

$$\vec{d}_{\rm FRF} = A\left(A^\dagger A\right)^{-1} A^\dagger \vec{d} = AA^\dagger \vec{d}, \tag{7}$$

where the second equality follows from the orthonormality of the DPSS modes, i.e. $A^\dagger A = I$.

HERA implements this least squares solution in the function `hera_filters.dspec._fit_basis_1d`, using the `scipy.optimize.lsq_linear` function. The A operator that contains the DPSS basis functions is generated by the function `hera_filters.dspec.dpss_operator`. Time flags are included in the solve, i.e. missing time samples are accounted for in this method.

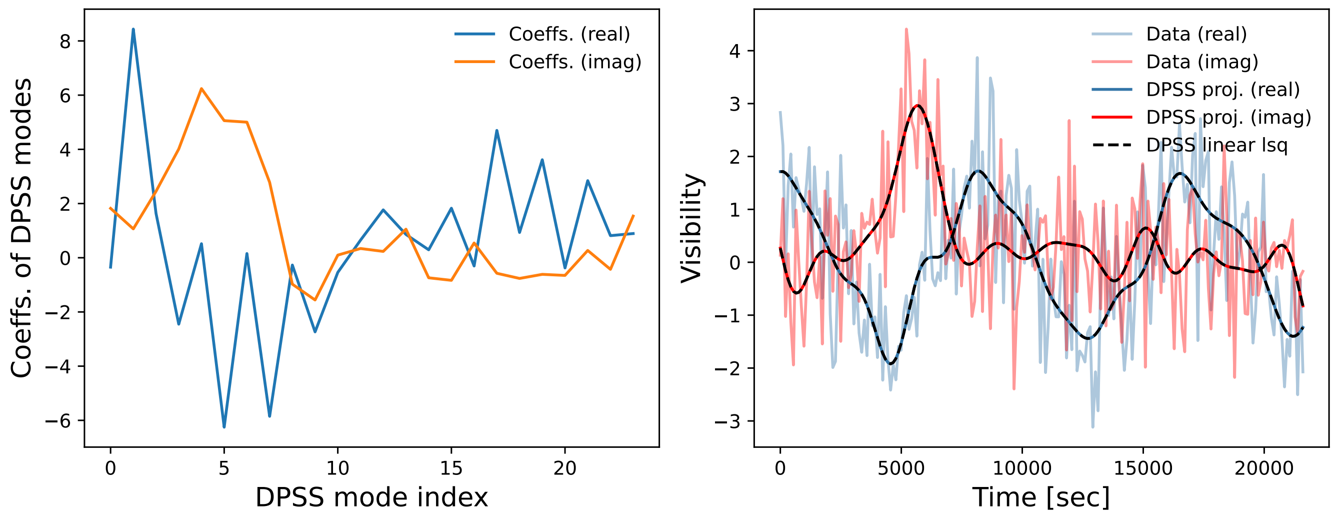

Figure 2: *(Left panel):* Real and imaginary coefficients of 24 DPSS modes that were fit to some made-up data consisting of arbitrary smooth functions plus complex white noise, for the same tophat filter as in Fig. 1. *(Right panel):* The made-up data (faint lines) with the filtered solution overplotted (darker lines). The FR filter is centred around $f = 0$ mHz, so is a low-pass filter; the filtered lines are clearly smoother, as expected.

Example application – The filter defined for Fig. 1 is a low-pass filter, centred on a fringe rate $f = 0$ mHz, with half-width $\Delta f = 0.5$ mHz. As an illustration, we have applied this filter to some made-up complex data that is a combination of smooth functions of time plus complex white noise, shown in the right-hand panel of Fig. 2. There are no flags/missing data. We applied the filter manually, using a simple projection $AA^\dagger \vec{d}$ (red/blue solid lines) and the `scipy` linear least-squares function (black dashed lines).

To generate the A matrix, we used the `hera_filters.dspec.dpss_operator` function with keyword argument `eigenval_cutoff` set to `1e-2`. This argument was used to determine how many DPSS modes to include in the matrix operator. 24 modes were included in total, which is significantly less than the total number of time samples in the data (200). Clearly, only structure that is smooth in time has survived the filtering, as expected.

Fig. 3 shows power spectra of the filtered and unfiltered data. The grey area centred around $f = 0$ mHz marks the tophat filter region. In this instance, the DPSS filter has quite accurately recovered the structure of the data at low delay, with a long tail of increasing power suppression outside the tophat filter region. A strict tophat filter would perfectly match the unfiltered data inside the grey region, and be exactly zero outside the grey region.

Selecting the number of modes to retain – The `dpss_operator` function provides several ways of selecting how many DPSS modes to retain to perform the filtering. Retaining more modes makes for a sharper transition around the boundary of the tophat region, but the higher-order modes also have lower concentration ratios.

The bottom panel of Fig. 3 shows how the filtered power spectrum changes as more DPSS modes are included. Retaining only one mode gives excellent suppression of power outside the tophat region, but loses a substantial fraction of the signal within the tophat region too, with even the centre of the filter (at $f = 0$ mHz) showing significant suppression. As more modes are retained, the recovery of the signal within the tophat region improves rapidly, with the centre of the filter being recovered better than the edges. The suppression in the region outside the tophat rapidly degrades however, as higher-order modes are less concentrated and so necessarily add power outside the tophat. Comparing with the fringe-rate profiles in Fig. 1, we also see that sufficiently high-order modes start to show features outside the tophat window, causing artifacts in the power spectrum immediately outside the tophat region.

In our example, $N \simeq 25$ is sufficient to accurately recover the signal in the tophat region right up to the tophat boundary. This comes at the cost of there being much less suppression of power outside the tophat window, to the point that at some fringe rates, the 'filtered' signal is actually above the input signal. The signal power outside the

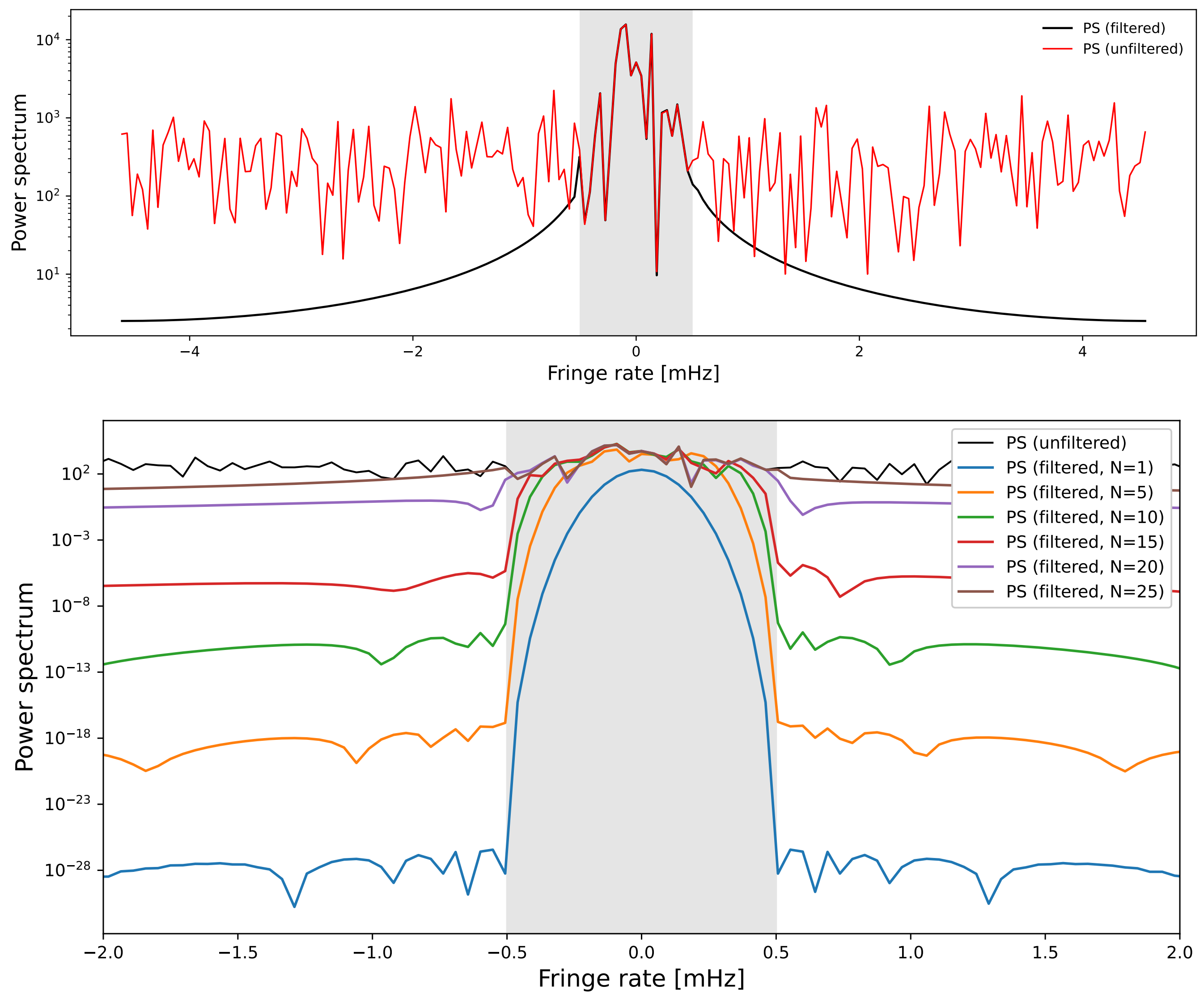

Figure 3: Fringe-rate power spectra of the unfiltered and filtered data. The shaded area shows the tophat filter region. The upper panel shows the result for the example described above that has 24 terms. The lower panel shows the result for filters with different numbers of terms.

tophat is very unlike the input data, and so could be interpreted as an error component rather than a filtered version of the input signal (see the next section for a discussion).

Several options are available to automatically determine how many modes to return based on the tophat that is being applied:

- `nterms`: Set the number of modes to retain manually.
- `eigenval_cutoff`: Do a cut on the eigenvalues of the matrix used to generate the DPSS modes. This should be equivalent to doing a cut on the concentration ratio of the modes.
- `edge_suppression`: Chooses the number of modes based on how much suppression would be achieved to a tone at the edge of the filter.
- `avg_suppression`: Similar to `edge_suppression`, but based on an average of the suppression for all the tones inside the filter instead.

Power outside the tophat region – Note that the DPSS filter (as implemented here) is not acting as a straightforward multiplicative fringe-rate filter in the region outside the tophat. The power in the filtered signal that is seen in Fig. 3 does not have a similar shape to the input data in this region of fringe-rate space. Instead, we can interpret the power outside the tophat window as an artifact caused by the 'tails' of the DPSS modes that are being used to fit the signal inside the window. These tails get heavier with increasing order of the DPSS functions, and eventually the highest order DPSS modes become more concentrated outside the window than inside. Retaining more DPSS modes tends to increase the amplitude of the filtered signal outside the window, as seen in Fig. 3.

Focusing on the case where only lower-order DPSS modes are retained, these tails give the filtered signal outside the window a sort of 'non-local' property – the amplitude of the low-order DPSS modes inside the window sets the amplitude of the filtered signal outside the window, while it is relatively insensitive to what the actual input signal outside the window is doing.

Acknowledgments – Thanks to Hugh Garsden, Bobby Pascua, and Mike Wilensky for useful comments.