LARGE-SCALE POWER LOSS IN GROUND-BASED CMB MAPMAKING

SIGURD NAESS
Institute of Theoretical Astrophysics, University of Oslo, Norway

THIBAUT LOUIS
Université Paris-Saclay, CNRS/IN2P3, IJCLab, 91405 Orsay, France

ABSTRACT

CMB mapmaking relies on a data model to solve for the sky map, and this process is vulnerable to bias if the data model cannot capture the full behavior of the signal. We demonstrate that this bias is not just limited to small-scale effects in high-contrast regions of the sky, but can manifest as $O(1)$ power loss on large scales in the map under conditions and assumptions realistic for ground-based CMB telescopes. This bias is invisible to simulation-based tests that do not explicitly model them, making it easy to miss. We identify two different mechanisms that both cause suppression of long-wavelength modes: sub-pixel errors and detector gain calibration mismatch. We show that the specific case of subpixel bias can be eliminated using bilinear pointing matrices, but also provide simple methods for testing for the presence of large-scale model error bias in general.

1. INTRODUCTION

CMB telescopes observe the sky by scanning their detectors across it while continuously reading off a series of samples from the detectors. Typically the signal-to-noise ratio of each sample is small, but by combining a large number of samples with knowledge of which direction the telescope was pointing at any time, it’s possible to reconstruct an image of the sky. There are several ways of doing this, with the most common being maximum-likelihood, filter+bin and destriping. These all start by modeling the telescope data as (Tegmark 1997)

$$d = Pm + n$$

where $d$ is the set of samples read off from the detectors (often called the time-ordered data), $m$ is the set of pixels of the sky image we want to reconstruct, $n$ is the noise in each sample (usually with significant correlations), and $P$ is a pointing matrix that encodes how each sample responds to the pixels in the image.

Given this data model, it’s possible to either directly solve for an unbiased map (as in maximum likelihood mapmaking or destriping), or to measure and correct for the bias in a biased estimator (as in filter+bin mapmaking). In the case of maximum-likelihood mapmaking, we solve for the least-squares solution

$$\hat{m} = (P^T N^{-1} P)^{-1} P^T N^{-1} d$$

Linear regression like this is optimal and unbiased as long as only the dependent variable ($d$ in this case) is noisy, but it is relatively well known to be subject to attenuation bias if the independent variables ($P$ in this case) are also noisy (Draper & Smith 1998). This is effectively what happens when we have model errors: The $P$ we assume deviates from the true behavior of the instrument $P_{true}$ by some perturbation which effectively acts as a noise source. Since $P$ enters into the denominator $(P^T N^{-1} P)$ in the second power, this noise term is squared, resulting in a noise bias there. We therefore end up dividing by too much, leading to $m$ being underestimated.

Similar things happen in destriping and filter+bin mapmaking for much the same reason: at some point, whether in the map-making or power spectrum estimation steps, one needs to divide out the bias introduced by filtering, and if there’s a noise bias in what’s being divided, then the result will be biased low.

In the context of CMB mapmaking, this bias has, to the extent that one has been aware of it at all, been thought
of as mainly a small-scale effect relevant only in regions of the sky with very high contrast, leading to artifacts like thin stripes or bowling around bright sources (Piazzo 2017; Naess 2019), or as a tiny correction on intermediate scales (Poutanen et al. 2006) saw a bias of 0.6% peaking at $\ell = 800$ for simulations of the Planck space telescope). The goal of this paper is to demonstrate the unintuitive and surprising result that that these effects can lead to $O(1)$ bias on large scales everywhere in the map. The full scope of these effects appears to be largely unappreciated, and we fear that some published results may suffer from uncorrected bias at low multipoles in total intensity due to these effects. As we shall see, it’s mainly ground-based telescopes that are vulnerable to this bias, which usually manifests as a power deficit at large scales, as illustrated in figure 1.

In the following we will go through two common cases of model error in detail. First subpixel errors, which arise when the smooth sky is approximated as a mosaic of pixels; and then detector gain mismatch, which is becoming increasingly important with today’s $O(10^5)$-detector telescopes. We will demonstrate the effects with 1D and 2D toy examples and derive analytic expressions for the bias; and test various mitigation methods such as reducing the filtering of correlated noise.

## 2. Subpixel Errors

Subpixel errors may be both the most common and most important class of model errors in CMB mapmaking. For efficiency reasons $P$ is almost always chosen to use simple nearest-neighbor interpolation, where the value of a sample is simply given by the value of the pixel nearest to it. This means that $P$ can be implemented by simply reading off one pixel value per sample, and its transpose $P^T$ consists of simply summing the value of the samples that hit each pixel. However, this comes at the cost of there being a discontinuous jump in values as one scans from one pixel to the next, as illustrated in figure 2. Hence, the closest the model can get to a smooth curve is a staircase-like function that hugs it, leaving a residual full of discontinuous jumps (the blue curve in the figure).

Discontinuous residuals are not necessarily problematic. The trouble arises when this is coupled with a likelihood where some modes have much less weight than others. Ground-based CMB telescopes suffer from atmospheric emission that acts as a large source of noise on long timescales. This leads to time-ordered data noise power spectra similar to the one sketched in figure 3, with a white noise floor at short timescales (high frequency) transitioning to a steep rise of several orders of magnitude as one moves to longer timescales (low frequency). In this case long-wavelength modes have orders of magnitude lower weight in the likelihood than short-wavelength modes. Put another way, they are orders of magnitude cheaper to sacrifice when the model can’t fully fit the data.

### 2.1. 1D Toy Example

To illustrate the interaction between a nearest neighbor pointing matrix’s subpixel errors and a likelihood where large scales have low weight, let us consider a simple 1D case where 100 samples scan uniformly across 10 pixels:\(^3\)

$$\text{n pix} = 10$$

$$\text{n samp} = 100$$

$$\text{pix} = \text{np.arange(n samp).astype(float)*n pix/n samp}$$

A standard nearest-neighbor pointing matrix for this looks like:

$$P = \text{np.zeros((n samp, n pix))}$$

for i, p in enumerate(range(n samp)):

$$P[i, \text{int(np.round(pix[i]))%n pix}] = 1$$

We assume a typical Fourier-diagonal “1/f” noise model $N(f) = 1 + (f/f_{\text{knee}})^\alpha$ with $f_{\text{knee}} = 0.03$ and $\alpha = -3.5$, and build the inverse noise matrix/filter/baseline-prior $F$ by projecting the inverse noise spectrum $N^{-1}$ into pixel space:\(^4\)

$$\text{freq} = \text{np.fft.rfftfreq(n samp)}$$

$$\text{inv_ps} = 1/(1+(\text{np.maximum(freq,freq[1]/2)}/0.03)$$

$$**-3.5)$$

$$F = \text{np.zeros((n samp,n samp))}$$

$$I = \text{np.eye(n samp)}$$

for i in range(n samp):

$$F[:,i] = \text{np.fft.irfft(inv_ps*np.fft.rfft(I[i]), n=n samp)}$$

The signal itself consists of just a long-wavelength sine wave:\(^5\)

$$\text{signal} = \text{np.sin(2*np.pi*pix/n pix)}$$

With this in place, we can now define our map estimators.

### 2.1.1. Binning

The binned map is simply the mean value of the samples in each pixel, with no weighting:

$$\text{map_binned = np.linalg.solve((P.T.dot(P)), P.T.dot(signal))}$$

This is the unweighted least-squares solution for the map. With a nearest neighbor pointing matrix where each sample only affects one pixel $P^TP$ is diagonal, making solving for the binned map very fast.

\(^1\) While methods with non-discontinuous pointing matrices exist, e.g. Keihänen & Reinecke (2012), these have had very limited adoption. As far as we are aware, every CMB telescope project has used nearest neighbor mapmaking for their official maps. This includes destripers like MADAM (Keihänen et al. 2010) and SRoll (Planck Collaboration 2020) used in Planck, the filter+bin map-makers used in SPT (Schaffer et al. 2011; Dutcher et al. 2021) and BICEP (BICEP2 Collaboration 2014) and the maximum likelihood map-makers used in ACT (Aiola et al. 2020) and QUlET (Ruud et al. 2015).

\(^2\) The equivalent for filter+bin is a filter that impacts some modes more than others (which is the whole point of a filter), and for destriping it’s the baseline length and any amplitude priors.

\(^3\) A real survey might have $\sim 3$ samples per pixel per scan for a single detector, but would end up with much more than that across multiple scans and multiple detectors.

\(^4\) In practice the “1/f” power law does not continue to infinity at zero frequency, but stabilizes at some high value at low frequency. We implement this by replacing $f = 0$ with a small, non-zero number for the purposes of computing the noise spectrum in the following.

\(^5\) Since all the mapmaking methods we will discuss are linear, signal and noise can analysed independently. Since the focus in this example is bias, it’s therefore enough to consider a signal-only data set, and we thus don’t add any noise.
2.1.2. Maximum-likelihood

The maximum-likelihood solution of equation 1 for the sky image \( m \) is

\[
\hat{m} = (P^T N^{-1} P)^{-1} P^T N^{-1} d
\]  

(3)

where \( N \) is the covariance matrix of the noise \( n \). This is the generalized least-squares solution for the map. Unlike binned mapmaking, the matrix \( P^T N^{-1} P \) except for the special case of uncorrelated noise. Solving for the maximum-likelihood map can therefore be hundreds of times slower, and requires iterative methods for large maps.\(^6\) In our toy example, \( N^{-1} = F \), so the maximum-likelihood map is

\[
\text{map}_{ml} = \text{nplinalg.solve}(P \cdot F \cdot F \cdot P) \cdot P \cdot F \cdot (signal))
\]

2.1.3. Filter+bin

As the name suggests, filter+bin consists of filtering the time-ordered data, and then making a binned map. We’ll use \( F \) as our filter, so the filter+bin map is

\[
\text{map}_{fb} = \text{nplinalg.solve}(P \cdot P \cdot F \cdot (signal))
\]

The filter+bin map is biased by design, so to interpret or debias it one needs to characterize this bias. There are two common approaches to doing this: Observation matrix and simulations.

**Observation matrix**— The observation matrix approach recognizes that the whole chain of operations observe, filter, map together make up a linear system, and can therefore be represented as a matrix, called the observation matrix (BICEP2 and Keck Array Collaboration 2016). Building this matrix is heavy, but doable for some surveys. Under the standard assumption that the observation step is given by equation 1, the observation matrix is given by

\[
\text{obsmat} = \text{nplinalg.inv}(P \cdot F \cdot F \cdot P)
\]

and using it, we can define a debiased filter+bin map

\[
\text{map}_{fb deobs} = \text{nplinalg.solve}(obsmat, \text{map}_{fb})
\]

\(^6\) The matrix \( P^T N^{-1} P \) is often poorly conditioned, making it slow for iterative solution methods to recover the large scales. If iteration is stopped early, this could result in a lack of power at large scales. This well-known phenomenon is not the bias we’re exploring in this paper, and since we solve the equation exactly it does not appear here.
Simulations—Alternatively, and more commonly, one can characterize the bias by simulating the observation of a set of random skies, passing them through the filter+bin process, and comparing the properties of the input and output images (e.g. Sayre et al. 2020). The standard way of doing this is by assuming that equation 1 describes the observation process, and that the bias can be described by a transfer function: a simple independent scaling of each Fourier mode. Under these assumptions, we can measure and correct the bias as follows.

\[
\text{nsim} = 1000 \\
\text{sim_ips} = \text{np.random.standard_normal(npix)} \\
\text{sim_ops} = \text{np.zeros(npix/2+1)} \\
\text{for i in range(nsim):} \\
\quad \text{sim_imap} = \text{np.linalg.solve(P.T.dot(P), P.T.dot(F).dot(P).dot(sim_imap))} \\
\quad \text{sim_ips += np.abs(np.fft.rfft(sim_imap))**2} \\
\quad \text{sim_ops += np.abs(np.fft.rfft(sim_ops))**2} \\
\text{map_fb_detrans = np.fft.irfft(np.fft.rfft(map_fb)/tf, n=npix)}
\]

2.1.4. Destriping

Destriping splits the noise into a correlated and uncorrelated part, and models the correlated noise as a series of slowly changing degrees of freedom to be solved for jointly with the sky image itself (Sutton et al. 2010; Tristram et al. 2011). The data is modeled as

\[
d = Pm + Qa + n_w \tag{4}
\]

where \(n_w\) is the white noise with diagonal covariance matrix \(N_w\), and \(Q\) describes how each correlated noise degree of freedom \(a\) maps onto the time-ordered data, typically in the form of seconds (ground) to minutes (space) long baselines. Given this, the maximum-likelihood solutions for \(a\) and \(m\) are

\[
Z = I - P(P^T N^{-1} P)^{-1} P^T N^{-1} \\
a = (Q^T N^{-1} ZQ + C_a^{-1} Q^T N^{-1} Zd)^{-1} \tag{5}
\]

where \(C_a\) is one’s prior knowledge of the covariance of \(a\). Similarly to binned and filter+bin mapmaking, destriping derives its speed from the diagonality of \(P^T N^{-1} P\) allowing for instant inversion. On the other hand, the equation for the baseline amplitudes \(a\) is not diagonal, and destriping allows a speed/optimality tradeoff in the choice of baseline length and prior. It approaches the maximum likelihood map when the baseline length is a single sample and \(C_a + N_w = N\). We implement this limit below, but explore other choices in section 2.2. In our toy example \(N_w = I\), so \(C_a = F^{-1} - I\).

\[
iCa = \text{np.linalg.inv(np.linalg.inv}(F) - I) \\
Z = I - P\text{.dot(np.linalg.solve(P.T.dot(P), P.T))} \\
a = \text{np.linalg.solve}(Z+iCa, Z.dot(signal)) \\
\text{map_ds = np.linalg.solve(P.T.dot(P), P.T.dot(signal - a))}
\]

2.1.5. Results

Figure 4 compares the recovered 1D sky “images” for the different mapmaking methods for this toy example. All methods are expected to have a small loss of power at small scale (called the “pixel window”) due to averaging the signal within each pixel, but this effect is well-known, easy to model, and not our focus here. We deconvolve it using the following function before plotting.

\[
def \text{devin}(x): \quad \text{return np.fft.irfft(np.fft.rfft(x)} / \text{np.sinc(freq), n=len(x))}\]

After pixel window deconvolution the binned map (green) closely matches the input signal (red). The same cannot be said for the other estimators. Maximum likelihood, filter+bin with observation matrix debiasing and destriping (which are all equivalent in the limit we consider here) are strongly biased, with the signal only being recovered with 1/3 of its real amplitude.

The situation is even worse for filter+bin with simulation-based debiasing, as this suffers from an additional bias due to assuming that the Fourier modes are independent.\(^7\)

2.1.6. Explanation

To see why inaccuracies in modeling the signal at sub-pixel scales can bias the largest scales in the map, let us consider the example in figure 2, where a detector measures a smooth, large-scale signal while moving across a few pixels. With a nearest-neighbor pointing matrix it is impossible to model this smooth signal: the model for each sample is simply that of the closest pixel, regardless of where inside that pixel it is. The model therefore looks like a staircase-like function in the time domain.

Given that we can’t exactly match the signal, what is the best approximation? Let us consider two very

\(^7\) This additional bias disappears if the simulations have exactly the same statistical properties as the real signal we wish to recover.
different alternatives. Model A: The value in each pixel is the average of the samples that hit it, making the model curve trace the smooth signal as closely as it can. This is the green curve in figure 2, and has the sawtooth-like residual shown with the blue curve. It is probably the model most people would choose if asked to draw one manually. Model B: The value in each pixel is zero, and the residual is simply the signal itself. Model B seems like a terrible fit to the data, but under a reasonable noise model for a ground-based telescope, like the one shown in figure 3, it will actually have a higher likelihood (lower $\chi^2 = r^2 N^{-1} r$ where $r$ is the residual) than model A. The reason is that while model B has a much larger residual than model A, model B’s residual is smooth and hence has most of its power at low frequency in the time domain where $N^{-1}$ is very small. Meanwhile, model A’s residual extends to all frequencies due to its jagged nature, including the costly high frequencies. The actual maximum-likelihood solution will be intermediate between these two cases, sacrificing some but not all of the large-scale power in the model to make the residual smoother.

To demonstrate that the bias really is caused by sub-pixel errors, we repeated the simulation with a signal that follows the same nearest-neighbor pixelization as the models. In this case all models except FB detrans are unbiased.

2.2. 2D toy example

The 1D toy example is useful for understanding the origin of the bias, but its unrealistic observing pattern makes it insufficient for exploring optimality/bias tradeoffs in the different methods. We therefore made a larger toy example where a single detector scans at constant speed across a square patch, sampling $N_{\text{scan}} = 400$ equi-spaced rows with $N_{\text{scan}}$ equi-spaced samples per row, followed by a column-wise scan the same patch, leading to a total cross-linked data set $N_{\text{samp}} = 2N_{\text{scan}}^2 = 3200$ samples long. This equi-spaced grid of samples was chosen to make it easy to simulate a signal directly onto the samples without needing the complication of pixel-to-sample projection. These samples will then be mapped onto a square grid of pixels with a side length of $N_{\text{side}} = 100$ using the different mapping methods.

For the signal we draw realizations from a CMB-like $1/f^2$ power spectrum with a Gaussian beam with standard deviation $\sigma = 3$ pixels. Here $l$ is the pixel-space wavenumber, which we evaluate in the higher resolution sample space as

$$
\begin{align*}
    l_x &= \text{np.fft.fftfreq}(N_{\text{scan}})[:,None] \ast N_{\text{side}}/N_{\text{scan}} \\
    l_y &= \text{np.fft.fftfreq}(N_{\text{scan}})[None,:] \ast N_{\text{side}}/N_{\text{scan}} \\
    l &= (l_y**2+l_x**2)**0.5
\end{align*}
$$

With this we can define the signal power spectrum $C_l = l^{-2}$ and beam $B_l = \text{exp}(-l^2\sigma^2/2)$, and draw signal realizations as

$$
\begin{align*}
    \text{signal\_map} &= \text{np.fft.ifft2}(\text{np.fft.fft2}(\text{np.random.standard_normal}((N_{\text{scan}},N_{\text{scan}})))\ast Cl**0.5*B1).
    \text{real} \\
    \text{signal\_told} &= \text{np.concatenate}([[\text{signal\_map}\_\text{reshape}(-1), \text{signal\_map}\_\text{T}\_\text{reshape}(-1)])
\end{align*}
$$

The last step here takes into account that the simulated scanning pattern covers the field twice, first horizontally and then vertically.

For the noise we use a simple $1/f$ spectrum $N(f) = 1 + (f/f_{\text{knee}})^\alpha$, with $f = \text{np.fft.fftfreq}(N_{\text{samp}})$ and the knee frequency $f_{\text{knee}}$ corresponding to $1/30$th of the side length, $f_{\text{knee}} = 0.5 \cdot 30/N_{\text{scan}} = 0.0375$ (6.7 pixel wavelength), and with an atmosphere-like exponent $\alpha = 3.5$.

The pixel coordinates of each sample are

$$
\begin{align*}
    \text{pix\_pat1} &= (\text{np.mgrid}([N_{\text{scan}},:N_{\text{scan}}])\ast N_{\text{side}}/N_{\text{scan}}).\text{reshape}(2,-1) \\
    \text{pix\_pat2} &= \text{pix\_pat1}[::-1] \\
    \text{pix} &= \text{np.concatenate}([[\text{pix\_pat1}, \text{pix\_pat2}],1])
\end{align*}
$$

which we use to build the nearest-neighbor pointing matrix

$$
\begin{align*}
    \text{ly, ix} &= \text{np.floor}(\text{pix}).\text{astype}(\text{int})\ast N_{\text{side}} \\
    \text{P\_nn} &= \text{scipy.sparse.car_array}((\text{np.full}(\text{N\_samp} ,1),(\text{np.arange}(\text{N\_samp}),\text{ly}\ast N_{\text{side}}\ast\text{ix})),\text{shape} = (\text{N\_samp},\text{N\_pix}))
\end{align*}
$$

2.2.1. Cases

We will investigate 5 classes of nearest-neighbor maps:

1. **bin**: Simple binned map, which we expect to be unbiased but very noisy

2. **ML**: Maximum-likelihood map. Ideally unbiased and optimal, but will deviate from this due to model errors.

3. **ML w:Chi**: Maximum-likelihood maps with a “whitened” noise model, which overestimates the white noise power by a factor $10^X$, $X \in \{1, 2, 3\}$, reducing the overall correlativeness of the noise model. We expect the bias to be proportional to the overall dynamic range of the noise model, so making the noise model whiter should reduce bias. The cost will be suboptimal noise weighting, leading to a noisier map, but it might be worth it.

4. **DS X**: Destriping map with a baseline of $X \in \{4, 16, 64\}$ samples and no amplitude prior. This is probably the most common type of destriping. The
Fig. 6.— Example signal and noise maps for the different mapmaking methods. The top left map is the input signal, which is directly evaluated at 4x higher resolution than what is used for reconstructing the output maps. All output maps were built from the same data and assume a nearest neighbor pointing matrix, except for the last row where a bilinear pointing matrix was used. The color range is the same for all panels. The binned map (top right) is bias-free but has uselessly high noise. Maximum likelihood (ML) and destriping with short baselengths (with (DS+) and without (DS) baseline amplitude prior, which only matters for the shortest baselines) are all low-noise but biased on large scales. This bias goes away as the noise model is artificially whitened (ML wX) or the baseline length is increased (for destriping), but this comes at a cost of increased noise on all scales. Bilinear mapmaking (last row) instead avoids the bias by eliminating most of the model error. It can be difficult to judge how significant the bias and noise levels are from these images. See figures 7 and 8 for easier to interpret comparisons of the signal and noise power spectra respectively.
Fraction of power retained

100
101
102
0.2
0.4
0.6
0.8
1.2
0
1
0.1
0.2
0.3
0.4
0.5

Wavenumber (l)

Results

Example maps are shown in figure 6, while the bias and noise are quantified in figures 7 and 8 respectively. As expected the binned map is unbiased but extremely noisy. The maximum-likelihood map is low-noise, but measurably biased on all scales, with a power deficit of a few percent at the smallest scales which grows to almost 100% on the largest scales. The deficit appears to fall proportionally with the whitening, with ML w1 and w2 being respectively 10x and 100x as close to unbiased. Sadly this comes at the cost of 40% and 350% higher noise power respectively. These numbers may differ for real-world cases, but this still seems like a very expensive bias mitigation method. All but the longest-baseline destriped maps are also strongly biased, with the shortest baseline being considerably worse than ML for almost all scales. Much like we saw with the ML variants, the less biased destriping versions come at a high cost in noise. Finally, the filter+bin map is biased even after simulation-based debiasing due to the simulations not capturing the subpixel behavior of the real data. Both the bias and noise levels are the same as maximum-likelihood in this example.

To test whether the observed biases are truly caused by subpixel errors, we repeat the simulations with only one sample per pixel ($N_{\text{scan}} = N_{\text{side}}$). As expected this results in an unbiased power spectrum for all methods.

2.2.3. Effective mitigation of subpixel errors

There will be no subpixel errors in the limit of infinitely small pixels, so it’s tempting to simply reduce the pixel size to solve the problem. This does work, but since subpixel errors are proportional to $|V_s \cdot \Delta |$, where $s$ is the true, smooth signal on the sky and $\Delta = [\Delta_x, \Delta_y]$ is the pixel shape, the improvement is only first order in the pixel side length. A much more feasible solution is to instead improve the subpixel handling in the pointing matrix. Going from nearest neighbor to bilinear interpolation is enough to practically eliminate subpixel errors without reducing pixel size. We can implement this in the toy example by defining

```
pix_left = np.floor(pix).astype(int)
ry, rx = pix-pix_left
iy1,ix1 = pix_left % N_side
iy2,ix2 = (pix_left+1) % N_side
weights = np.concatenate((
    (1-ry)*(1-rx), (1-ry)*rx,
    ry *(1-rx), ry *rx))
samps = np.tile(np.arange(Nsamp),4)
```

8 Even filter+bin, which ignores mode coupling, ends up having an unbiased power spectrum because the simulations used to build the transfer function followed the same distribution as the real signal.
The bias is also greatly reduced for destriping, but not line of pixels. Both detectors point in the same direction, noise. We will illustrate this using a minimal 1D toy model correlates observations with inconsistent gain, for few times slower.

maximum-likelihood mapmaking, which only becomes a them here in order to investigate the effect on the bias, like binning, filter+bin and destriping rely on for their
detectors (unweighted) mapmaking. Square it to get the effect on the noise covariance is diagonal in Fourier space

Bilinear mapping results in a different pixel window than the standard pixwin_nn = sinc(ly)[[:,None]*sinc (lx)[None,]) of nearest neighbor mapmaking. We derive this in appendix A, and the result is pixwin_lin = linwinid(ly)[[:,None]*linwinid(lx)[None,], linwinid(l1)= sinc(l1)**2/(1/3*(2-cos(2*pi*l)))]. This is plotted in figure 9.

Since each sample in bilinear mapmaking gets contributions from the four closest pixels, the pointing matrix is about four times slower than the standard nearest neighbor version. However, as shown in figures 7 and 8 this cost is worth it, with bilinear maximum-likelihood mapmaking being bias-free and even lower noise than standard ML. The bias is also greatly reduced for destriping, but not eliminated.

Note that bilinear mapmaking (and in general any mapmaking method where each sample affects multiple pixels) breaks the diagonality of $P^T P$ which methods like binning, filter+bin and destriping rely on for their speed. While we have implemented bilinear variants of them here in order to investigate the effect on the bias, bilinear filter+bin and bilinear destriping are too slow to be useful in practice. This is in contrast to bilinear maximum-likelihood mapmaking, which only becomes a few times slower.

3. GAIN ERRORS

Surprisingly, gain errors can also lead to a scale-dependent power loss. This happens when the noise model correlates observations with inconsistent gain, for example multiple miscalibrated detectors with correlated noise. We will illustrate this using a minimal 1D toy example with two correlated detectors, as well as a more realistic 2D example.

3.1. 1D toy example

Consider two miscalibrated detectors scanning across a line of pixels. Both detectors point in the same direction, and take one sample per pixel. We can model this as

$$d = G(Pm + n) = GPm + Gn \quad \text{(actual)}$$

where $d$ is the vector of samples, $m$ is the true signal in the pixels, $n$ is Gaussian noise with covariance $N$, $P$ is the pointing matrix and $G$ is the gain error. Written out in terms of detectors and samples, we can express this as

$$d_{di} = G_dP_d m_i + G_d n_{di}$$

with $d$ and $i$ being the detector and sample index respectively, $P_d = [1,1]_d = 1$ and $G_d = [g_1,g_2]_d$. This simply expresses that both detectors see the same signal at the same time. Let us assume that the noise covariance is diagonal in Fourier space

$$N_{d_{di}d_{dj}} = A_f C_{f_{di}f_{dj}} \delta_{ff'}$$

where $f$ and $f'$ are frequency indices, and $A_f$ is the noise power spectrum. This gives us the inverse noise model

$$N^{-1}_{d_{di}d_{dj}} = A_f^{-1} C^{-1}_{f_{di}f_{dj}} \delta_{ff'}$$

The maximum-likelihood solution of equation 6 is

$$\hat{m} = (P^T G^{-1} N^{-1} G^{-1} P)^{-1} P^T G^{-1} N^{-1} G^{-1} d$$

which has expectation value $\langle \hat{m} \rangle = m$, which is unbiased. However, given that $G$ is supposed to be a gain error, we are presumably not aware of it, and so our data model will instead be

$$d = Pm + Gn \quad \text{(model)}$$

The reason why $G$ is still present in front of $n$ in the model is that the noise properties are almost always measured from the data, so our noise model would automatically absorb $G$. For example, if a detector were to be calibrated too high, it would also appear noisier, and would therefore be downweighted more under inverse variance weighting. Given this model, our maximum-likelihood solution for $m$ is

$$\hat{m} = (P^T G^{-1} N^{-1} G^{-1} P)^{-1} P^T G^{-1} N^{-1} G^{-1} d$$

Inserting the actual data behavior from equation 6, we get

$$\langle \hat{m} \rangle = (P^T G^{-1} N^{-1} G^{-1} P)^{-1} P^T G^{-1} N^{-1} G^{-1} G P m$$

$$= (P^T G^{-1} N^{-1} G^{-1} P)^{-1} P^T G^{-1} N^{-1} P m$$

Since $N$ is diagonal in Fourier space and both $G$ and $P$ are constant in time, the whole equation becomes Fourier-diagonal

$$\langle \hat{m}_f \rangle = (P^T G^{-1} N_f^{-1} G^{-1} P)^{-1} P^T G^{-1} N_f^{-1} P m_f$$

Inserting eq. 10, this simplifies to ($A_f$ cancels)

$$\langle \hat{m}_f \rangle = g_1 g_2 \frac{g_1 + g_2}{g_1^2 + g_2^2} \frac{1 - \alpha_f}{2 g_1 g_2} \alpha_f m_f$$
So the effect of the gain miscalibration is an overall scaling of the result as one would expect, \(g_1 g_2 (g_1 + g_2) / (g_1^2 + g_2^2)\), times a transfer function

\[
TF(f) = \left(1 - \frac{1 - \alpha f}{1 - \frac{2 g_1 g_2}{g_1^2 + g_2^2} \alpha f}\right)^{-1}
\]

This reduces to 1 both when the detectors are uncorrelated \((\alpha f = 0)\) and when there are no relative gain errors \((g_1 = g_2)\).

To motivate a frequency-dependent correlation coefficient, let’s consider the common case of atmospheric \(1/f\)-noise plus white instrumental noise. Our detectors are co-pointing, meaning they see the same atmosphere, so this noise component is 100% correlated between detectors, but we assume their instrumental noise to be uncorrelated.

\[
N_f = \left[\begin{array}{c} 1 \\ 1 \\ 0 \end{array}\right] (f/f_{\text{knee}})^{-3} + \left[\begin{array}{c} 0 \\ 1 \\ 0 \end{array}\right] = \left[\begin{array}{c} 1 \\ \alpha f \\ 1 \end{array}\right] A_f
\]

\[
\alpha f = 1 - A_f^{-1}, \quad A_f = 1 + (f/f_{\text{knee}})^{-3} \Rightarrow
\]

\[
TF(f) = \frac{A_f^{-1}}{1 - \frac{2 g_1 g_2}{g_1^2 + g_2^2} (1 - A_f^{-1})}
\]

The result is a frequency-dependent correlation coefficient that changes smoothly from 100% correlated for atmosphere-dominated scales (low frequencies) to 0% correlated for detector-noise dominated scales (high frequencies). The transfer function for this is plotted in figure 10 for the case of a 10% gain error and average weather (tough note that this 1D example does not capture the benefits of crosslinking).

---

3.2. 2D toy example

We have also demonstrated the effect of gain errors on a somewhat more realistic 2D simulation, where an array of four detectors scans first horizontally and then vertically across a grid of 200 \times 200 pixels, with one sample per pixel and each sample hitting the pixel center to avoid subpixel errors.

3.3. Time-dependent gain errors are much less serious

So far we have only considered cases where detectors disagree on the gain, but gain errors can also be time-

---

\[^9\text{In filter+bin mapmaking the equivalent to reducing the correlations in the noise model would be to filter less.}\]
We further assume as we shall see the answer is “yes, but only in unrealistic circumstances”. Consider a single detector that makes a series of passes over a line of pixels, such that

\[ d_{i,x} = g_{i,x}m_{x,i} + g_{i,x}n_{i,x} \quad \text{(actual)} \quad (20) \]

where \( d_{i,x} \) is the measurement in pixel/sample \( x \) of pass \( i \), \( m_{x,i} \) is the true signal in pixel \( x \), \( g_{i,x} \) is the unmodeled gain and \( n_{i,x} \) is the noise. We assume that the gain error is ignored in the data model but captured by the noise model:

\[ d_{i,x} = m_{x,i} + g_{i,x}n_{i,x} \quad \text{(model)} \quad (21) \]

This has maximum-likelihood solution

\[ \sum g_{i,x}^{-1}N_{y,x}^{-1}g_{i,y}^{-1} \hat{m}_y = \sum g_{i,x}^{-1}N_{y,x}^{-1}(m_y + n_{i,y}) \quad (22) \]

We further assume

1. Both the noise and gain errors are independent between each pass

2. The inverse gain perturbation \( \gamma_{i,x} = g_{i,x}^{-1} - 1 \) has stationary covariance \( \Gamma \), which is therefore diagonal in Fourier space, with diagonal \( \hat{\Gamma} \) (its power spectrum).

3. Similarly the covariance is Fourier-diagonal with power spectrum \( \hat{N} \)

Under these assumptions it can be shown (see appendix B) that when the number of passes is much greater than one the Fourier transform of the maximum likelihood solution becomes

\[ \hat{m}_f = \frac{N_f^{-1}}{N_f^{-1} + S^{-2}(\Gamma \ast \hat{N})^{-1}_f} \hat{m}_f \quad (23) \]

where * is the convolution operator and \( S \) is the number of pixels, and we’ve assumed the standard discrete Fourier transform normalization. We see that in the limit of uncorrelated gain errors, \( \Gamma_f = \text{const} \), the bias becomes \( 1/(1 + \text{const} \cdot \hat{\Delta}) \). For atmospheric noise this gives a strong suppression of the signal on large scales, just like we saw for detector errors. On the other limit where the gain error changes slowly and is approximately constant over a noise correlation length, we have \( \Gamma \approx \text{const} \cdot \delta \Rightarrow \Gamma \ast \hat{N} \approx \text{const} \cdot \hat{N} \), and so the bias just becomes a constant scaling. So to get scale-dependent power loss, the gain error needs to vary on time-scales over which the noise is strongly correlated.

However, all of this rests on the assumption that the gain errors are captured by the noise model. For the previous case of constant per-detector gain errors this is realistic - after all detectors typically don’t all have the same noise level, so they need to be weighted differently, and the noise model ends up absorbing the gain error when being measured from the data itself. The situation is different from time-dependent gains. As we saw above, the noise model must be modulated by gain errors that change quickly relative to the atmosphere if we are to get scale-dependent bias, and for a typical ground-based CMB telescope this works out to be \( \sim \) second time-scales. It is very uncommon to build noise models that track the noise properties with a time resolution this high.

In contrast to detector-dependent gain errors, we therefore do not expect time-dependent gain errors to be a significant source for large-scale power loss.

4. EASILY MISSED IN SIMULATIONS

What makes model error bias especially insidious is that it is completely invisible to any simulation that does not explicitly include that particular type of model error. For example, a simulation where the CMB is read off from a simulated map using the same pointing matrix as is used in the mapmaking itself would be blind to subpixel errors. And simulations that don’t inject gain errors into both the data and noise model will be blind to power loss from gain errors. Given the many possible ways the real data might deviate from one’s model of it, it is hard to be sure that one has included all the relevant types of model error in the simulations. It is therefore very easy to trick oneself into believing that one has an unbiased analysis pipeline while there are in fact \( O(1) \) biases remaining.

5. DETECTION STRATEGIES

The unintuitive large-scale effects of model errors originate from the interaction between model errors and nonlocal weighting/filtering. A noise model (or filter) with a large dynamic range, such as one capturing the huge signal-to-noise ratio between the long-wavelength and short-wavelength noise power typical for ground-based CMB telescopes, is therefore much more vulnerable to large-scale power loss than one appropriate for space-based telescopes which have almost flat noise power spectra. This suggests the following tests for large-scale model error bias:
1. Compare power spectra with those from a space-based telescope. This is reliable but comes at the cost of being able to make an independent measurement.

2. Split the data into subsets with different ratios of correlated noise to white noise and check their consistency. This could for example be a split by the level of precipitable water-vapor (PWV) in the atmosphere. High-PWV-data would be expected to have a higher dynamic range and therefore be more vulnerable.

3. Map the same data both using the standard noise model/filter and a less optimal one an artificially reduced lower dynamic range, e.g. one that underestimates the amount of correlated noise or ignores correlations between detectors. The latter should result in less a biased but noisier map, as we saw in figures 7 and 8. If the maps are consistent, then large-scale model error bias is probably not an issue. Alternatively, one can check for consistency when varying the pixel size. We haven’t tested the effectiveness of this method, but model error bias should be proportional to the pixel size with nearest neighbor mapmaking.

6. WHO IS AT RISK?

While model error bias can be expected to always be present at a low level, it only becomes important when dealing with very large amounts of correlated noise (e.g. much more noise correlations on some scales than others, see equation 19). Therefore, the telescopes at risk are those that fulfill the following criteria.

1. Observe from the ground.

2. Care about total intensity, since the atmosphere is mostly unpolarized.

3. Observe at frequencies with high water vapor emission, since water vapor is the main source of clumpiness in the atmospheric emission. At CMB-relevant frequencies, the lower the frequency the safer. Frequencies < 60 GHz should be relatively safe while > 200 GHz are particularly vulnerable.

4. Have sensitive detectors, since lower white noise means a larger ratio between this noise and the correlated noise from the atmosphere.

5. Have a large number of detectors with a low angular separation on the sky, since this increases their correlatedness.

There aren’t that many telescopes that fulfill these criteria, as ground-based telescopes have focused mostly on polarization or lower frequencies. After reviewing LAMBDA’s list of CMB experiments we found four > 60 GHz ground-based CMB telescope projects started after year 2000\(^\text{10}\) with published TT spectra for \(\ell < 2000\): ACBAR, ACT, BICEP and SPT. Of these, ACBAR, ACT (Choi et al. 2020) and SPT-3G (Balkenhol et al. 2022) cut their low-\(\ell\) data, indicating problems recovering these scales. Meanwhile BICEP (BICEP1 Collaboration 2013; BICEP2 and Keck Array Collaboration 2018) and SPT-pol (Henning et al. 2018) publish TT spectra down to \(\ell < 100\) with no signs of problems. The only case where we could see clear evidence for large-scale power-loss is in the multipoles just above the multipole cutoff in ACT, as seen in figure 13 (reproduced from Choi et al. (2020)).

---

\(^{10}\) Ground-based projects older than this are too insensitive for this bias to be visible even if it were to be present.

---

1. All current CMB mapmaking methods are vulnerable to model error bias, including maximum-likelihood mapmaking, destriping and filter+bin mapmaking.

2. Subpixel errors and gain errors are common examples of model errors, with subpixel errors being almost universal. Subpixel bias is caused by the likelihood disfavoring the sharp edges implied by the pixelization, causing it to discard low-S/N signal.
Gain errors enter into inverse variance weighting squared, leading to a noise bias term that dominates strongly downweighted/filtered scales. Other types of model error can also be important.

3. Model error bias can manifest in unintuitive ways, with a common symptom being a large (O(1)) loss of long-wavelength power in the maps.

4. The size of the bias is proportional to the dynamic range of the filter/noise model. Hence it is important for ground-based measurements of the unpolarized CMB due to the presence of large-scale atmospheric noise there, but is much less important for polarization measurements or space-based telescopes.

5. Simulations are blind to these biases unless specifically designed to target them. There is a large risk of ending up thinking one has an unbiased pipeline despite there being large bias in the actual CMB maps.

6. An effective way of testing for this bias is to also map the data using a (much) lower dynamic range noise model/filter\(^{11}\) and checking if this leads to consistent power spectra.

While it’s unclear how large a role model-error bias has played in ground-based CMB results published so far, it will be an increasingly important effect as sensitivity increases. We hope upcoming projects like Simons Observatory and CMB-S4 will be on the lookout for large-scale power loss in total intensity, and to be aware that simulations alone cannot prove that their analysis pipeline is bias free.

ACKNOWLEDGEMENTS

We would like to thank Jon Sievers for first making us aware that subpixel errors weren’t just limited to X’s around point sources and other small-scale effects, and Ed Wollack and Lorenzo Piazza for useful comments. Yunyang Li pointed out an error in the original version of the 1D gain toy example (now fixed). This work was supported by a grant from the Simons Foundation (CCA 918271, PBL).

REFERENCES

Aiola, S., et al. 2020, arXiv:2007.07288, J. Cosmology Astropart. Phys., 2020, 047, The Atacama Cosmology Telescope: DR4 maps and cosmological parameters

Balkenhol, L., et al. 2022, arXiv:2212.05642, arXiv e-prints, arXiv:2212.05642, A Measurement of the CMB Temperature Power Spectrum and Constraints on Cosmology from the SPT-3G 2018 TT/TE/EE Data Set

BICEP1 Collaboration. 2013, arXiv:1310.1422, Degree-Scale CMB Polarization Measurements from Three Years of BICEP1 Data

BICEP2 and Keck Array Collaboration. 2016, arXiv:1603.05976, ApJ, 825, 66, BICEP2/Keck Array: VI. Matrix Based E/B Separation Applied to Bicep2 and the Keck Array

—. 2018, arXiv:1810.05216, Phys. Rev. Lett., 121, 221301, Constraints on Primordial Gravitational Waves Using Planck, WMAP, and New BICEP2/Keck Observations through the 2015 Season

BICEP2 Collaboration. 2014, arXiv:1403.3985, Phys. Rev. Lett., 112, 241101, Detection of B-Mode Polarization at Degree Angular Scales by BICEP2

Choi, S. K., et al. 2020, arXiv:2007.07289, J. Cosmology Astropart. Phys., 2020, 045, The Atacama Cosmology Telescope: a measurement of the Cosmic Microwave Background power spectra at 98 and 150 GHz

Draper, N. R. & Smith, H. 1998, Applied Regression Analysis, 3rd edn., Wiley Series in Probability and Statistics (Nashville, TN: John Wiley & Sons)

Dutcher, D., et al. 2021, arXiv:2101.01604, Phys. Rev. D, 104, 022003, Measurements of the E-mode polarization and temperature-E-mode correlation of the CMB from SPT-3G 2018 data

Henning, J. W., et al. 2018, arXiv:1707.09353, ApJ, 852, 97, Measurements of the Temperature and E-mode Polarization of the CMB from 500 Square Degrees of SPTpol Data

Keihänen, E., Keenikalo, R., Kurki-Suonio, H., Poutanen, T., & Sirviö, A. S. 2010, arXiv:0907.0367, A&A, 510, A57, Making cosmic microwave background temperature and polarization maps with MADAM

Keihänen, E., & Reinecke, M. 2012, arXiv:1208.1399, A&A, 548, A110, ArtDeco: a beam-deconvolution code for absolute cosmic microwave background measurements

Naess, S. K. 2019, JCAP, 2019, 060-060, How to avoid X’s around point sources in maximum likelihood CMB maps, http://dx.doi.org/10.1088/1475-7516/2019/12/060

Piazza, L. 2017, ITIP, 26, 5232, Least Squares Image Estimation for Large Data in the Presence of Noise and Irregular Sampling Planck Collaboration. 2020, arXiv:1807.06207, A&A, 641, A3, Planck 2018 results. III. High Frequency Instrument data processing and frequency maps

Poutanen, T., et al. 2006, astro-ph/0601504, A&A, 449, 1311, Comparison of map-making algorithms for CMB experiments

Rudr, T. M., et al. 2015, arXiv:1508.02778, ApJ, 811, 89, The QU Imaging Experiment: Polarization Measurements of the Galactic Plane at 43 and 95 GHz

Sayre, J. T., et al. 2020, arXiv:1910.05748, Phys. Rev. D, 101, 122003, Measurements of B-mode polarization of the cosmic microwave background from 500 square degrees of SPTpol data

Schaffer, K. K., et al. 2011, arXiv:1111.7245, ApJ, 743, 90, The First Public Release of South Pole Telescope Data: Maps of a 95 deg² Field from 2008 Observations

Sutton, D., et al. 2010, arXiv:0912.2738, MNRAS, 407, 1387, Fast and precise map-making for massively multi-detector CMB experiments

Tagmark, M. 1997, AJ, 140, L87–L90, How to Make Maps from Cosmic Microwave Background Data without Losing Information, http://dx.doi.org/10.1086/310631

Tristram, M., Filliard, C., Perdereau, O., Plaszczynski, S., Stompor, R., & Touze, F. 2011, arXiv:1103.2281, A&A, 534, A88, Iterative destriping and photometric calibration for Planck-HFI, polarized, multi-detector map-making

11 Or much longer baselines for destriping
APPENDIX

A. LINEAR PIXEL WINDOW

Let us start by considering the 1D case. Given a set of pixels $\{i\}$ with values $\{v_i\}$ we define the linear interpolation value $d$ at some pixel coordinate $i + x$ where $x \in [0, 1)$ as

$$ d = (1 - x)v_i + xv_{i+1} $$

This is illustrated in figure 14.

![Figure 14](image)

Fig. 14.— Definition of the quantities involved in linear interpolation. Given values $\{v_i\}$ sampled at locations $\{i\}$, we define the linear interpolation value $d$ at some location $i + x$ where $x \in [0, 1)$ as $d = (1 - x)v_i + xv_{i+1}$.

We can generalize this to a set of samples $d_a$ with

$$ d_a = (1 - x_a)v_a^{\text{left}} + x_a v_a^{\text{right}} $$

where $v_a^{\text{left}}$ and $v_a^{\text{right}}$ are the values of the pixel to the left and right of where sample $a$ points, and $x_a$ is its subpixel position. This linear system can be written as the following matrix equation

$$ d = Pv $$

$P$’s form will depend on the exact scanning pattern, but the final pixel window should only depend on each pixel being hit equally much at all subpixel locations. For simplicity, we will assume a simple scanning pattern that scans a single time across a row of pixels from left to right, with a constant scanning speed and infinite sample rate. This results in the pointing matrix $P$ taking the form shown in figure 15.

![Figure 15](image)

Fig. 15.— Structure of a linear-interpolation pointing matrix $P$ and its transpose $P^T$ for the case of a set of samples making a single scan across a row of pixels in 1D, assuming wrapping boundary conditions. $x \in [0, 1)$ is the subpixel coordinate of each sample.

Given the set of samples $d$, the least squares estimator for the pixel values $v$ under uniform weighting is

$$ \hat{v} = (P^TP)^{-1}P^Td $$

The denominator $P^TP$ consists of two types of non-zero cases:

$$ (P^TP)_{00} = [\cdots 1 - x \cdots] \begin{bmatrix} \cdots & \cdots & \cdots \\ 1 - x & \cdots & \cdots \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \end{bmatrix} \begin{bmatrix} \cdots \\ x \\ \vdots \\ \vdots \\ \end{bmatrix} = S \int_0^1 (1 - x)^2 dx + S \int_0^1 x^2 dx = \frac{2}{3}S $$

(A5)

$$ (P^TP)_{01} = [\cdots 1 - x \cdots] \begin{bmatrix} \cdots & \cdots & \cdots \\ 1 - x & \cdots & \cdots \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \end{bmatrix} \begin{bmatrix} \cdots \\ x \\ \vdots \\ \vdots \\ \end{bmatrix} = S \int_0^1 (1 - x)xdx = \frac{1}{6}S $$

(A6)
Here \( S \) is the number of samples per pixel, and appears due to the summing implied by \( P^T \). \( S \to \infty \) in the integral limit we used here, but it will end up cancelling. With this, we end up with the denominator taking the Toeplitz form

\[
P^T P = \frac{1}{6} N \begin{bmatrix}
1 & 1 & 1 \\
1 & 4 & 1 \\
1 & 4 & 1 \\
\vdots \\
4 & 1 & 1 \\
1 & 4 & 1 \\
1 & 4 & 1 \\
\end{bmatrix}
\]  
(A7)

Toeplitz matrices are diagonal in Fourier space, so \((P^T P)^{-1}\) becomes a simple element-wise division there. The Fourier-diagonal is given by the Fourier transform of the first row of the Toeplitz matrix. Defining \( A = P^T P \), it’s Fourier diagonal as \( \tilde{a} \) is given by

\[
\tilde{a}(k) = \frac{1}{N} \sum_{j=0}^{N-1} e^{-\frac{2\pi i k j}{N}} A_{j0} = \frac{S}{N} \frac{1}{6} \left(4 + e^{\frac{2\pi i k}{N}} + e^{-\frac{2\pi i k}{N}}\right) = \frac{S}{N} \frac{1}{3} \left(2 + \cos \frac{2\pi k}{N}\right) 
\]  
(A8)

where \( N \) is the number of pixels, \( i \) is the imaginary unit, \( k \) is the zero-based index into \( \tilde{a} \) and \( j \) is the zero-based row index of \( A \).

Turning to the numerator \( b = P^T d \), we recognize that it is simply the convolution of \( d \) with the triangle function \( T(x) = (1 - |x|)S, x \in [-1, 1] \), evaluated at each pixel center.\(^\text{12}\) Using the convolution theorem, we can therefore directly write down

\[
\tilde{b}(k) = \tilde{T}(k) \cdot \tilde{d}(k) 
\]  
(A9)

We’re left with computing the Fourier-space representation of the triangle function \( T \).

\[
\tilde{T}(k) = \frac{S}{N} \int_{-1}^{1} e^{-\frac{2\pi i k x}{N}} (1 - |x|) dx
\]

\[
= \frac{S}{N} \left[ \int_{-1}^{1} e^{-\frac{2\pi i k x}{N}} dx - \int_{-1}^{0} e^{-\frac{2\pi i k x}{N}} x dx - \int_{0}^{1} e^{-\frac{2\pi i k x}{N}} (-x) dx \right]
\]

\[
= \frac{S}{N} \left[ 2 \sin \frac{2k}{N} - 2 \sin \frac{2k}{N} + 2 \left(\frac{2\pi k}{N}\right)^2 \left(1 - \cos \frac{2\pi k}{N}\right) \right]
\]

\[
= \frac{S}{N} \left( \sin \frac{k}{N} \right)^2 
\]  
(A10)

Combining the numerator and denominator we have

\[
\hat{v} = \frac{\tilde{b}(k)}{\tilde{a}(k)} \tilde{d}(k) = \left( \sin \frac{k}{N} \right)^2 \frac{1}{3 (2 + \cos \frac{2\pi k}{N})} \tilde{d}(k) = W_{\text{lin}}(k/N) \tilde{d}(k) 
\]

(A11)

where the linear interpolation pixel window \( W_{\text{lin}}(f) \) is defined as

\[
W_{\text{lin}}(f) = \frac{\sin(f)^2}{3 (2 + \cos(2\pi f))} \]  
(A12)

This pixel window is compared to the standard nearest neighbor pixel window \( W_{\text{nn}} = \sin(f) \) in figure 16. Their behavior is quite different. While \( W_{\text{nn}} \) falls monotonically to around 0.6 at the Nyquist frequency, \( W_{\text{lin}} \) rises to about 1.4 at \( f = 0.4 \), and only then starts falling.

The natural generalization of linear interpolation to two dimensions is bilinear interpolation, which is the composition of a horizontal and vertical one-dimensional interpolation. Because this operation is separable, so is the resulting pixel window. Hence

\[
W_{\text{lin}}^{2D}(f_y, f_x) = W_{\text{lin}}(f_y) W_{\text{lin}}(f_x) 
\]  
(A13)

\(^{12}\) This means that \( T \ast d \) is the same function as \( b \), just \( S \) times more densely sampled. Hence, it has the same Fourier representation up to \( b \)’s Nyquist frequency.
Fig. 16.— Comparison the 1D pixel windows for nearest neighbor mapmaking ($W_{nn}$) and linear mapmaking ($W_{lin}$), plotted up to the Nyquist frequency, which is $\frac{1}{2}$ in these units.

B. TIME-DEPENDENT GAIN ERRORS

Continuing from section 3.3, we had

$$A_{xy} \left[ \sum_{i} g_{ix}^{-1} N_{xy}^{-1} g_{iy}^{-1} \right] \hat{m}_y = \sum_{i} g_{ix}^{-1} N_{xy}^{-1} (m_y + n_{iy})$$  \tag{B1}$$

Expanding $g_{ix}^{-1} = 1 + \gamma_{ix}$ and assuming the detector makes $M$ passes over the pixels, we get

$$A_{xy} = \sum_{i} (1 + \gamma_{ix}) N_{xy}^{-1} (1 + \gamma_{iy})$$

$$= M \left( N_{xy}^{-1} + \frac{1}{M} \sum_{i} \gamma_{ix} N_{xy}^{-1} + \frac{1}{M} \sum_{i} N_{xy}^{-1} \gamma_{iy} + \frac{1}{M} \sum_{i} \gamma_{ix} N_{xy}^{-1} \gamma_{iy} \right)$$  \tag{B2}$$

In the limit $M \to \infty$ the odd powers of the random variable $\gamma$ disappear, so we are left with computing $\psi$. Since both $N$ and $\Gamma$ are diagonal in Fourier space, it makes sense to express $\psi$ in Fourier space. Denoting Fourier-space quantities with a tilde, and defining the Fourier transform operators

$$F_{kx} = \exp(-2\pi \sqrt{-1} kx / S)$$

$$F_{xk}^{-1} = \frac{1}{N} \exp(2\pi \sqrt{-1} kx / S)$$  \tag{B3}$$
where $S$ is the number of pixels, we get

$$\tilde{\psi}_{jk} = \frac{1}{M} \sum_{i} \sum_{xy} F_{jx}(F^{-1}\tilde{\gamma})_x (F^{-1}\tilde{N})_{xy} (F^{-1}\tilde{\gamma})^\dagger_y F^\dagger_{yk}$$

$$= \frac{1}{M} \sum_{i} \sum_{xy} F_{jx}(F^{-1}\tilde{\gamma})_x (F\tilde{N}^{-1}F)_xy (F^{-1}\tilde{\gamma})^\dagger_y F^\dagger_{yk}$$

$$= \frac{1}{M} \sum_{i} \sum_{xy\alpha\beta} \frac{1}{N^2} \exp \left( \frac{2\pi\sqrt{-1}}{S} \left( -jx + \alpha x + \beta x - \gamma y + ky \right) \tilde{\gamma}_\alpha \tilde{N}^{-1} \tilde{\gamma}^\dagger_\gamma \right)$$

$$= \sum_{\alpha} \left( \frac{1}{M} \sum_{i} \tilde{\gamma}_\alpha \tilde{N}^{-1}_{j-\alpha} \tilde{\gamma}^\dagger_{\alpha} \tilde{N}^{-1}_{\alpha} \right)$$

$$= \sum_{\alpha} \left( \frac{1}{M} \sum_{i} \tilde{\gamma}_\alpha \tilde{N}^{-1}_{j-\alpha} \delta_{jk} \right)$$

$$= \left( \tilde{\Gamma} \ast \tilde{N}^{-1} \right) \delta_{jk} \quad \text{(B4)}$$

So $\psi$ is Fourier diagonal just like $\Gamma$ and $N^{-1}$, and its diagonal is the convolution of their diagonals. With this we can complete the expression for $A$.

$$\tilde{A} = FAF^\dagger = F \left( (F^{-1}\tilde{N}F^{-1})^{-1} + F^{-1}\tilde{\psi}F^{-1} \right) F^\dagger = MS^2 \left( \tilde{N}^{-1} + S^{-2}\tilde{\Gamma} \ast \tilde{N}^{-1} \right) \quad \text{(B5)}$$

Moving on to the right-hand-side of the equation for $\tilde{m}$, we have

$$b_x = \sum_{i} (1 + \gamma_{ix}) N^{-1}_{xy} (m_y + n_{iy})$$

$$= M \left( N^{-1}_{xy} m_y + \frac{1}{M} \sum_{i} \gamma_{ix} N^{-1}_{xy} n_{iy} \right) \quad \text{(odd)}$$

$$= M \left( \frac{1}{M} \sum_{i} \gamma_{ix} N^{-1}_{xy} m_y + \frac{1}{M} \sum_{i} \gamma_{ix} N^{-1}_{xy} n_{iy} \right) \quad \text{(odd)}$$

All but the first term fall out if we assume that the fluctuations in $\gamma$ and $n$ are independent. Going to Fourier space, we are left with

$$\tilde{b} = Fb = MF \left( F^{-1}\tilde{N}F^{-1} \right) ^{-1} F^{-1}\tilde{m} = M F F^\dagger \tilde{N}^{-1} F F^\dagger \tilde{m} = MS \tilde{N}^{-1} \tilde{m} \quad \text{(B7)}$$

With this the full Fourier-space equation for $A\tilde{m}$ becomes:

$$A\tilde{m} = b \Leftrightarrow F^{-1}\tilde{A} \tilde{N}^{-1} F^{-1}\tilde{b} \Leftrightarrow \tilde{m} = \frac{\tilde{N}^{-1}}{\tilde{N}^{-1} + S^{-2}\tilde{\Gamma} \ast \tilde{N}^{-1}} \tilde{m}$$

The factor $S^{-2}$ is an artifact of the Fourier normalization used, and would differ for other Fourier unit choices.