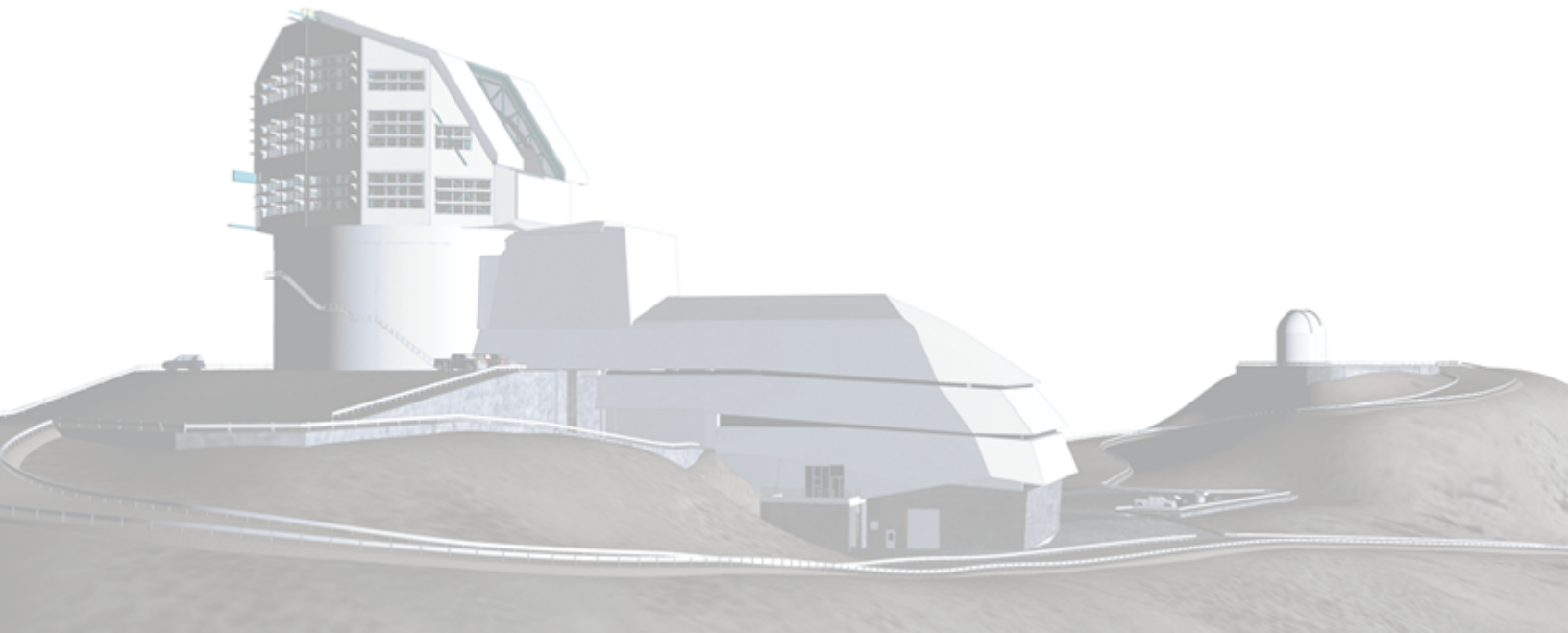

Vera C. Rubin Observatory
Data Management

# Cell-Based Coaddition

**Jim Bosch**

**DMTN-075**

**Latest Revision: 2021-10-06**

## Abstract

Design sketch and mathematics for a new approach to building coadds and constraining their inputs.

# Change Record

| Version | Date | Description | Owner name |
|---|---|---|---|
| 1 | YYYY-MM-DD | Unreleased. | Jim Bosch |

*Document source location:* `https://github.com/lsst-dm/dmtn-075`

# Contents

# Cell-Based Coaddition

## 1 Likelihood Coadds

### 1.1 Derivation and starting assumptions

We start by writing the log likelihood of an arbitrary model of the local sky, $\alpha$, given multiple images $z_j$ with differing PSFs $\phi$:

$$L \equiv \frac{1}{2} \sum_j \boldsymbol{R}_j^T \boldsymbol{C}_j^{-1} \boldsymbol{R}_j \tag{1}$$

where the residual vectors[1] $\boldsymbol{R}_j$ are defined as

$$R_j[\boldsymbol{x}_j] \equiv z_j[\boldsymbol{x}_j] - \int d^2\boldsymbol{r}\, \phi_j\left(\boldsymbol{A}_j \boldsymbol{x}_j + \boldsymbol{b}_j - \boldsymbol{r}\right)\, \alpha(\boldsymbol{r}) \tag{2}$$

where $\boldsymbol{r}$ are coordinates in the projection we will coadd onto, and $\boldsymbol{A}_j$ and $\boldsymbol{b}_j$ are the linear transform and translation that map input-image coordinates $\boldsymbol{x}_j$ to $\boldsymbol{r}$. Note that we define the PSF and model in coadd coordinates, not input-image coordinates. This likelihood is *local* because we have assumed the PSF is not spatially varying and the transformation is affine – a valid assumption in the neighborhood of any particular point, but not over larger scales. This is a common feature of the approaches to coaddition described here: we will explicitly subdivide the sky into small cells whose maximum extent is set by the scales on which these assumptions are valid.

This likelihood is quite general in many respects:

- the input pixel grid may or may not be well-sampled;
- pixels on the same image may have arbitrary noise correlations;
- pixels with no data or artifacts can simply be omitted from the sum, provided they can be identified.

[1] When we use matrix notation on image-like entities, we will invariably mean that the 2-d image is flattened into a 1-d vector.

We will need to place restrictions on these to make the method computationally tractable in later steps. In other respects, we have made powerful simplifying assumptions:

- all images have been background subtracted in advance;
- the true sky is completely static ($\alpha$ is the same for every epoch);
- there is no wavelength dependence at all - so either the true sky is monochromatic, or all PSFs have the same throughput as a function of wavelength, so all wavelength dependency cancels.

We will relax each of these assumptions in later sections.

We now expand the quadratic log likelihood and group by powers of $\alpha$:

$$L = \frac{\kappa}{2} - \int d^2\boldsymbol{r}\, \Psi(\boldsymbol{r})\, \alpha(\boldsymbol{r}) + \frac{1}{2} \int d^2\boldsymbol{r} \int d^2\boldsymbol{s}\, \Phi(\boldsymbol{r}, \boldsymbol{s})\, \alpha(\boldsymbol{r})\, \alpha(\boldsymbol{s}) \tag{3}$$

with

$$\kappa_c \equiv \sum_j \kappa_j \equiv \sum_j \sum_{\boldsymbol{x}_j} \sum_{\boldsymbol{y}_j} C_j^{-1}[\boldsymbol{x}_j, \boldsymbol{y}_j]\; z_j[\boldsymbol{x}_j]\, z_j[\boldsymbol{y}_j] \tag{4}$$

$$\Psi_c(\boldsymbol{r}) \equiv \sum_j \Psi(\boldsymbol{r}) \equiv \sum_j \sum_{\boldsymbol{x}_j} \sum_{\boldsymbol{y}_j} C_j^{-1}[\boldsymbol{x}_j, \boldsymbol{y}_j]\; z_j[\boldsymbol{x}_j]\, \phi_j\left(\boldsymbol{A}_j \boldsymbol{y}_j + \boldsymbol{b}_j - \boldsymbol{r}\right) \tag{5}$$

$$\Phi_c(\boldsymbol{r}, \boldsymbol{s}) \equiv \sum_j \Phi(\boldsymbol{r}, \boldsymbol{s}) \equiv \sum_j \sum_{\boldsymbol{x}_j} \sum_{\boldsymbol{y}_j} C_j^{-1}[\boldsymbol{x}_j, \boldsymbol{y}_j]\; \phi_j\left(\boldsymbol{A}_j \boldsymbol{x}_j + \boldsymbol{b}_j - \boldsymbol{r}\right)\; \phi_j\left(\boldsymbol{A}_j \boldsymbol{y}_j + \boldsymbol{b}_j - \boldsymbol{s}\right) \tag{6}$$

$$\tag{7}$$

$\Psi_c$ and $\Phi_c$ are sufficient statistics for our model ($\kappa_c$ does not constrain the model; it enters only if one is interested in the Bayesian evidence or other goodness-of-fit measures), and they include all sums over input images. But as continuous functions, we can't consider them a coadd, let alone a practical one.

If we choose our output grid such that all input PSFs are well sampled, however, we can replace

$$\phi_j\left(\boldsymbol{A}_j\boldsymbol{x}_j + \boldsymbol{b}_j - \boldsymbol{r}\right) = \sum_{\boldsymbol{p}} \phi_j\left(\boldsymbol{A}_j\boldsymbol{x} + \boldsymbol{b}_j - \boldsymbol{p}\right)\ \mathrm{sinc}^2(\boldsymbol{p} - \boldsymbol{r}) \tag{8}$$

and obtain

$$\Psi_c(\boldsymbol{r}) = \sum_{\boldsymbol{p}} \Psi_c(\boldsymbol{p})\ \mathrm{sinc}^2(\boldsymbol{r} - \boldsymbol{p}) \tag{9}$$

$$\Phi_c(\boldsymbol{r}, \boldsymbol{s}) = \sum_{\boldsymbol{p}} \sum_{\boldsymbol{q}} \Phi_c(\boldsymbol{p}, \boldsymbol{q})\ \mathrm{sinc}^2(\boldsymbol{r} - \boldsymbol{p})\ \mathrm{sinc}^2(\boldsymbol{s} - \boldsymbol{q}) \tag{10}$$

i.e. $\Psi_c$ and $\Phi_c$ are guaranteed to be well-sampled as well, even if the input pixel grids are not. We are still quite far from a practical coadd, however; we want an image as well as a PSF and/or covariance matrix that enter into the likelihood in essentially the same way that a single input image does. That means we want to solve

$$\Psi(\boldsymbol{r}) = \sum_{\boldsymbol{p}} \sum_{\boldsymbol{q}} C_c^{-1}[\boldsymbol{p}, \boldsymbol{q}]\, \phi_c[\boldsymbol{r} - \boldsymbol{p}]\, z_c[\boldsymbol{q}] \tag{11}$$

$$\Phi(\boldsymbol{r}, \boldsymbol{s}) = \sum_{\boldsymbol{p}} \sum_{\boldsymbol{q}} C_c^{-1}[\boldsymbol{p}, \boldsymbol{q}]\, \phi_c[\boldsymbol{r} - \boldsymbol{p}]\, \phi_c[\boldsymbol{s} - \boldsymbol{q}] \tag{12}$$

for $\phi_c$, $C_c$, and $z_c$. There is no single unique solution, and in fact no guarantee of any solution for completely general inputs. This approach is one limit of the `IMCOM` method of **?**, which utilizes a weight function that minimizes S/N losses during coaddition along with errors in PSF reconstruction; when there is no S/N loss and the PSF reconstruction is exact, their solution satisfies [11] and [12], while providing diagnostics when no such solution is possible. `IMCOM` is also extremely computationally expensive, as is any method that relies on direct accumulation of a fully general $\Phi$. In this technical note, we will instead focus on methods that start with simplifying assumptions that are at least *mostly* true, and then investigate ways to iteratively and incrementally deal with those assumptions being weakly violated.

The first of these simplifying assumptions is that the input noise in each image is constant:[2]

$$C_j[\boldsymbol{x}_j, \boldsymbol{y}_j] \to \sigma_j^2\, \delta[\boldsymbol{x}_j - \boldsymbol{y}_j] \tag{13}$$

[2] It should be possible to assume only that the noise is stationary, but the algebra is much more involved.

A single-input contribution to $\Phi_c$ is then

$$\Phi_j(\boldsymbol{r}, \boldsymbol{s}) = \frac{1}{\sigma_j^2} \sum_{\boldsymbol{x}_j} \phi_j\left(\boldsymbol{A}_j \boldsymbol{x}_j + \boldsymbol{b}_j - \boldsymbol{r}\right)\, \phi_j\left(\boldsymbol{A}_j \boldsymbol{x}_j + \boldsymbol{b}_j - \boldsymbol{s}\right) \tag{14}$$

with Fourier transform

$$\widetilde{\Phi_j}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{\sigma_j^2} \widetilde{\phi_j}(-\boldsymbol{u})\, \widetilde{\phi_j}(-\boldsymbol{v})\, e^{-2\pi i \boldsymbol{b}^T(\boldsymbol{u}+\boldsymbol{v})} \sum_{\boldsymbol{x}_j} e^{-2\pi i \boldsymbol{x}_j^T \boldsymbol{A}_j^T(\boldsymbol{u}+\boldsymbol{v})} \tag{15}$$

To simplify further, we'd like to be able to apply the Poisson summation formula to the sum over $\boldsymbol{x}_j$, but we need to be careful about the summation limits; so far we've implicitly (and vaguely) summed over all input pixels relevant to a cell, excluding any completely missing pixels. To instead sum over $\mathbb{Z}^2$, we need the original sum in [2] to truncate outside some finite region by defining both $z_j$ and $\alpha$ to be zero outside that region, and we need to assume there are no missing pixels. Neighboring cells will necessarily define truncation inconsistently from each other, so we will also need to build the coadd for a cell with some padding, such that coadd pixels within the cell are (to good approximation) not affected by the truncation, and thus consistent with their neighbors. How much padding will depend on the details of the algorithm, and the degree to which we aim for compactness in the real domain vs. compactness in the Fourier domain.

With these qualifications, $\widetilde{\Phi_j}$ becomes

$$\widetilde{\Phi_j}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{\sigma_j^2} \widetilde{\phi_j}(-\boldsymbol{u})\, \widetilde{\phi_j}(-\boldsymbol{v})\, e^{-2\pi i \boldsymbol{b}^T(\boldsymbol{u}+\boldsymbol{v})} \sum_{\boldsymbol{k}_j \in \mathbb{Z}^2} \delta\left(\boldsymbol{k}_j - \boldsymbol{A}_j^T(\boldsymbol{u}+\boldsymbol{v})\right) \tag{16}$$

But $\phi_j$ is band-limited on the coadd pixel grid, so $\widetilde{\phi_j}$ is zero if $|\boldsymbol{u}|_1 \geq \frac{1}{2}$ or $|\boldsymbol{v}|_1 \geq \frac{1}{2}$, and we can assume $|\boldsymbol{u}+\boldsymbol{v}|_1 < 1$. Considering the case where $\boldsymbol{A}_j$ is a uniform scaling for simplicity:

- if $|A_j| < 1$, the coadd pixels are larger than the input pixels; only the $\boldsymbol{k}_j = 0$ term in the sum survives (note that the input image then must be well-sampled, because the coadd is).
- if $|A_j| > 1$, the coadd pixels are smaller than the input pixels, and other terms may survive only if $A_j$ is greater than twice the actual band limit of the PSF – or, in other words, only if the input image is not well sampled.

Our final simplifying assumption is naturally that the input images are well-sampled; this yields

$$\widetilde{\Phi_j}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{\sigma_j^2} \widetilde{\phi_j}(-\boldsymbol{u})\, \widetilde{\phi_j}(-\boldsymbol{v})\, e^{-2\pi i \boldsymbol{b}^T(\boldsymbol{u}+\boldsymbol{v})}\, \delta\left(-\boldsymbol{A}_j^T(\boldsymbol{u}+\boldsymbol{v})\right) \tag{17}$$

$$= \frac{1}{\sigma_j^2} \left|\widetilde{\phi_j}(\boldsymbol{u})\right|^2 \delta(\boldsymbol{u}+\boldsymbol{v}) \tag{18}$$

If we then accumulate the contributions from each input image in the Fourier domain, the delta function factors out of the sum, and we can write

$$\widetilde{\Phi_c}(\boldsymbol{u}, \boldsymbol{v}) = \delta(\boldsymbol{u}+\boldsymbol{v}) \sum_j \frac{1}{\sigma_j^2} \left|\widetilde{\phi_j}(\boldsymbol{u})\right|^2 = \delta(\boldsymbol{u}+\boldsymbol{v}) \frac{1}{\sigma_c^2} \left|\widetilde{\phi_c}(\boldsymbol{u})\right|^2 \tag{19}$$

This is essentially the continuous Fourier version of [12], but here the solution is trivial, and in many respects ideal:

$$\widetilde{\phi_c}(\boldsymbol{u}) = \sigma_c \sqrt{\widetilde{\Phi_c}(\boldsymbol{u}, \boldsymbol{u})} = \sigma_c \sqrt{\sum_j \frac{1}{\sigma_j^2} \left|\widetilde{\phi_j}(\boldsymbol{u})\right|^2} \tag{20}$$

$$\frac{1}{\sigma_c} = \int d^2\boldsymbol{u} \sqrt{\sum_j \frac{1}{\sigma_j^2} \left|\widetilde{\phi_j}(\boldsymbol{u})\right|^2} \tag{21}$$

Note that

- the coadd PSF $\phi_c$ is the Fourier-domain square root of $\Phi_c$;
- the noise in the coadd is also constant, and its standard deviation $\sigma_c$ can be derived as the constant the normalizes the coadd PSF $\phi_c$.

With these in hand, we can compute the Fourier-domain coadd image $z_c$ from the Fourier transform of $\Psi_c$:

$$\widetilde{z_c}(\boldsymbol{u}) = \frac{\widetilde{\Psi_c}(\boldsymbol{u})\, \sigma_c^2}{\widetilde{\phi_c}(\boldsymbol{u})} \tag{22}$$

Obtaining $\widetilde{\Psi_c}$ from $z_j$ and $\phi_j$ involves a continuous Fourier transform of either noisy, sampled data or something derived from it; a subtly difficult task. But it is also possible to reframe this

as a linear operation in the image domain:

$$z_c(\boldsymbol{r}) = \sum_j \sum_{\boldsymbol{x}_j} K_j\left(\boldsymbol{r} - \boldsymbol{A}_j\boldsymbol{x}_j - \boldsymbol{b}_j\right)\ z_j[\boldsymbol{x}_j] \tag{23}$$

$$\widetilde{K}_j(\boldsymbol{u}) \equiv \frac{\sigma_c^2\,\widetilde{\phi_j^*}(\boldsymbol{u})}{\sigma_j^2\,\widetilde{\phi_c}(\boldsymbol{u})} \tag{24}$$

This simple result still hides a lot of practical complexity, because $K$ is

- generally not compact in the image domain (in fact, it is band-limited, and hence *formally* must have infinite extent in the image domain);
- potentially expensive to evaluate at the many non-integer points implied by 23.

Overall, it is unclear whether we would be best served by approach that approximates the continuous Fourier transforms with DFTs (with all of the boundary-condition and folding issues that always entails) or one that works in the image domain, perhaps with analytic approximations or basis functions in the representation of $\phi$ that have analytic Fourier transforms. We will explore several possibilities in Section 2.

## 1.2 Approximate PSFs

In many case we may not want to use or may not be able to use the best available model of the PSF when building a coadd. Using an approximation formally breaks the property that the coadd is a sufficient statistic for the static sky, but the actual information loss could nevertheless be quite small (much smaller than the loss due to building naive direct coadds in all cases). In many of the implementation options we discuss later in Section 2, using an approximation with an analytic Fourier transform is required. In other cases:

- The approximation may be more compact in the Fourier and/or image domain.
- The approximation may be less affected by measurement noise or numerical instability, both of which can cause trouble with deconvolution operations.
- The true PSF may be chromatic, so even our best model evaluated with some nomimal SED is only approximately right for other SEDs.

- We may have already built the coadd with a PSF we now know to be subtly wrong, and want to understand systematic errors this may have introduced into later processing and/or update the coadd without starting over.

Because the coadd likelihood will no longer be exactly equivalent to the original joint single-epoch likelihood with this approximate, we will instead start with [23] as a prescription for building a coadd, but replace the true (or best-estimate) per-epoch PSFs $\phi_j$ with deliberate approximations $p_j$, and then derive the new effective coadd PSF.

We start by expanding $\widetilde{K}$ with that substition:

$$\widetilde{K}_j(\boldsymbol{u}) \equiv \frac{\sigma_c^2 \, \widetilde{p_j^*}(\boldsymbol{u})}{\sigma_j^2 \sqrt{\sum_n \frac{1}{\sigma_n^2} \left|\widetilde{p_n}(\boldsymbol{u})\right|^2}} \tag{25}$$

Note that $\sigma_c$ is at this stage just an unknown normalization constant; we will show later that it is equal to the noise on the coadd when the effective coadd PSF is normalized.

To compute the effective PSF of the coadd, we coadd the image of a point source centered at $\boldsymbol{r} = 0$ using [23]:

$$\phi_c(\boldsymbol{r}) = \sum_j \sum_{\boldsymbol{x}_j} K_j\left(\boldsymbol{r} - \boldsymbol{A}_j \boldsymbol{x}_j - \boldsymbol{b}_j\right) \, \phi_j\left(\boldsymbol{A}_j \boldsymbol{x}_j + \boldsymbol{b}_j\right) \tag{26}$$

It is important here that we use the true PSFs $\phi_j$ (or our best-available model). We are also continuing to rely on our assumption that all input images are well-sampled - without that, $K$ would need to be a position-dependent kernel to have a well-defined PSF on the coadd. That assumption also lets us perform this convolution in coadd coordinates[3]:

$$\phi_c(\boldsymbol{r}) = \sum_j \frac{1}{|A_j|} \sum_{\boldsymbol{s}} K_j(\boldsymbol{r} - \boldsymbol{s}) \, \phi_j(\boldsymbol{s}) \tag{27}$$

We can now solve for $\sigma_c$ by requiring

$$\int \phi_c(\boldsymbol{r}) \, d^2\boldsymbol{r} = 1 \tag{28}$$

[3] TODO: prove this by a Fourier round-trip; we get another sum where only $\boldsymbol{k} = 0$ survives.

The noise covariance of the coadd can be computed via straightforward uncertainty propagation:

$$C_c(\boldsymbol{r}, \boldsymbol{s}) = \sum_j \sigma_j^2 \sum_{\boldsymbol{x}_j} K_j\left(\boldsymbol{r} - \boldsymbol{A}_j \boldsymbol{x}_j - \boldsymbol{b}_j\right)\ K_j\left(\boldsymbol{s} - \boldsymbol{A}_j \boldsymbol{x}_j - \boldsymbol{b}_j\right) \tag{29}$$

Its Fourier transform is

$$\widetilde{C_c}(\boldsymbol{u}, \boldsymbol{v}) = \sum_j \sigma_j^2 \widetilde{K_j}(\boldsymbol{u})\, \widetilde{K_j}(\boldsymbol{v})\, e^{-2\pi i \boldsymbol{b}_j^T(\boldsymbol{u}+\boldsymbol{v})} \sum_{\boldsymbol{x}_j} e^{-2\pi i \boldsymbol{x}_j^T \boldsymbol{A}_j^T(\boldsymbol{u}+\boldsymbol{v})} \tag{30}$$

$$= \sum_j \sigma_j^2 \left|\widetilde{K_j}(\boldsymbol{u})\right|^2 \delta(\boldsymbol{u} + \boldsymbol{v}) \tag{31}$$

where the simplification comes from using the Poisson summation formula and rejecting modes other than $\boldsymbol{k} = 0$, just as in [18]. Substituting the definition of $\widetilde{K}$ from [25], this simplifies further to

$$\widetilde{C_c}(\boldsymbol{u}, \boldsymbol{v}) = \left[\sum_j \frac{\sigma_c^2}{\sigma_j^2} \left|\widetilde{p_j}(\boldsymbol{u})\right|^2\right] \left[\sum_n \frac{1}{\sigma_n^2} \left|\widetilde{p_n}(\boldsymbol{u})\right|^2\right]^{-1} = \sigma_c^2 \tag{32}$$

Even with the PSF approximated, the noise in the coadd is uncorrelated and stationary.

With this formalism in hand, we can now look more closely at how this coadd algorithm compares to those in broad use in astronomy. A "direct" coadd is built by simply resampling all input images to the same grid, and combining them with a weighted mean. The weights must be very slowly-varying (often a per-epoch constant), and the combination must use a linear operator (no means or sigma-clipping) for the coadd to have a well-defined effective PSF. When the weights are proportional to the exposure time, this approach is statistically equivalent to taking a single longer exposure (at least for the ideal, static sky), but it is not optimal when the input images have different PSFs. In fact, we can see here that it is equivalent to using a delta function for all PSF approximations $p_j$ (and kernels $K_j$), or a constant weight in the Fourier domain; the optimal coadd with $p_j \to \phi_j$ instead gives more weight to high spatial frequency contributions from the best-seeing images.

PSF-matched coadds, in which each input image is degraded to a target PSF that is approximately the same as the PSF of the worst-seeing image, are naturally even worse from a signal-to-noise standpoint. This makes sense in our formalism: these define $\widetilde{K_j}$ such that $\widetilde{p_j}$ appears in the denominator, not the numerator, and hence they downweight the high-frequency

modes from the better-seeing images even more than the constant Fourier weighting of direct coadds.

### 1.3 Full noise propagation

While the algorithm we have derived assumes stationary, uncorrelated noise, we can still use it coadd images with more complex noise if we are willing to ignore the structure of that noise when determining the weights. Just like PSF approximations, this means our coadds will no longer be a sufficient statistic for the static sky, but the loss of information should be quite small in practice.

To fully propagate the noise, we can use the same formula as [29], adjusted to use a full covariance matrix $\boldsymbol{C}_j$ for each input image:

$$C_c[\boldsymbol{r}, \boldsymbol{s}] = \sum_j \sum_{\boldsymbol{x}_j} \sum_{\boldsymbol{y}_j} K_j\left(\boldsymbol{r} - \boldsymbol{A}_j \boldsymbol{x}_j - \boldsymbol{b}_j\right)\, K_j\left(\boldsymbol{s} - \boldsymbol{A}_j \boldsymbol{y}_j - \boldsymbol{b}_j\right)\, C_j\left[\boldsymbol{x}_j, \boldsymbol{y}_j\right] \tag{33}$$

Unfortunately, even if the noise is just nonstationary (i.e. still uncorrelated), as is common in astronomical images, the noise on the coadd ends up both nonstationary and correlated, leaving [33] quite expensive in both compute and storage.

In practice, it will often be sufficient to propagate only the diagonal $C_c[\boldsymbol{r}, \boldsymbol{r}]$ (or even just the original constant approximation $\sigma_c$), and possibly augment this with a few Monte Carlo noise fields: noise images that are coadded in exactly the same way as the true input images. Even extremely demanding (in terms of systematic error control) science applications such as shear estimation for weak lensing can often get by with just one Monte Carlo noise realization (TODO citation).

## 2 Implementation options

### 2.1 Resampling in advance

In this section, we consider implementation options that assume all input images have already been resampled to the coadd grid in advance, presumably via a compact image-domain interpolant that approximates sinc (e.g. Lanczos). This inevitably introduces aliasing, which is

traditionally ignored; we will ignore it here as well, but do consider it to be one of the disadvantages of these approaches.

### 2.1.1 Native-resolution FFT convolution and deconvolution

In this approach, input data images $z_j$ are FFT'd, which means we must first zero-pad them and apply some sort of multiplicative ramp to a buffer region between the output data region and the zero region, because the presence of sharp edges would cause problems. Best-available PSF models $\phi_j$ and approximations $p_j$ are either rendered directly in the Fourier domain or rendered as images and also FFT'd onto the same (discrete) Fourier grid. We then accumulate the following sums:

$$\widetilde{\Psi_c}[\boldsymbol{u}] = \sum_j \frac{\widetilde{p_j^*}[\boldsymbol{u}]\,\widetilde{z_j}[\boldsymbol{u}]}{\sigma_j^2} \tag{34}$$

$$\widetilde{\Phi_c^{(1)}}[\boldsymbol{u}] = \sum_j \frac{\widetilde{p_j^*}[\boldsymbol{u}]\,\widetilde{\phi_j}[\boldsymbol{u}]}{\sigma_j^2} \tag{35}$$

$$\widetilde{\Phi_c^{(2)}}[\boldsymbol{u}] = \sum_j \frac{\widetilde{p_j^*}[\boldsymbol{u}]\,\widetilde{p_j}[\boldsymbol{u}]}{\sigma_j^2} \tag{36}$$

The unnormalized coadd PSF model $\widetilde{\phi_c}$ is then

$$\frac{\widetilde{\phi_c}[\boldsymbol{u}]}{\sigma_c^2} = \frac{\widetilde{\Phi_c^{(1)}}[\boldsymbol{u}]}{\sqrt{\widetilde{\Phi_c^{(2)}}[\boldsymbol{u}]}}\,; \tag{37}$$

we can then inverse FFT and solve for $\sigma_c$ via

$$\sum_{\boldsymbol{r}} \phi_c[\boldsymbol{r}] = 1\,. \tag{38}$$

Finally, the Fourier-domain coadd image is

$$\widetilde{z_c}[\boldsymbol{u}] = \frac{\widetilde{\Psi_c}[\boldsymbol{u}]\,\sigma_c^2}{\sqrt{\widetilde{\Phi_c^{(2)}}[\boldsymbol{u}]}}\,, \tag{39}$$

which we also inverse FFT to form the coadd image.

This approach is in many respects the simplest possible implementation, especially if one already has an optimized pipeline for image resampling. It is also probably among the fastest: there are no expensive image-domain convolutions, and the padding necessary for the FFT is *probably* only proportional to the PSF size, not the size of the coadd cell.

The main downside of this approach is its nonlocality in the image domain. The FFT division by $\sqrt{\widetilde{\Phi_c^{(2)}}[\boldsymbol{u}]}$ in particular combines information from all pixels when computing the value of any other pixel, so even output pixels far from the cell edge will be affected by our artificial multiplicative ramp to some degree. The same is true of bad or missing pixels that are interpolated in the input images; their effect cannot be confined to some region around the interpolation.

It may be that this is not a serious problem in practice; the operations may be "local enough" in the image domain for any systematic errors introduced to be negligible. But this is largely an experimental question, and it certainly depends on the profiles of the input PSFs and their approximations; the challenges come from the kids of messy situations that are often hard to simulate, let alone analyze mathematically.

Because this method never actually constructs the weighting operators $\boldsymbol{K}_j$ directly, there is no direct way to propagate a full noise covariance matrix with it via [33]; if that kind of noise propagation is desired one would instead have to construct a compact $\boldsymbol{K}_j$ using a different approach (e.g. Section 2.1.2) that is not quite the same as the one effectively used here. The same is true of the extensions in Section 3 that make use of an image-domain $\boldsymbol{K}_j$.

### 2.1.2 Single-kernel image-domain convolution

This method is in many respects the opposite of the previous one, though it also assumes the input data has been resampled. We work entirely in the image domain; we compute approximations to $\boldsymbol{K}_j$ and then use [23] and [27] to directly accumulate the coadd image and effective PSF.

In order to make these discrete convolution operations efficient, we need to define $\boldsymbol{K}_j$ approximations with strictly finite support in the image domain (even if it is derived from an analytic form that does not truncate, we need to render images from it that do). This directly addresses the locality problems with FFT methods, like the one described in Section 2.1.1;

zero-padding and artificial ramps are unnecessary. We do need to expand the region we convolve by a buffer the size of $K_j$, but after doing so we can be confident that the original target region is fully unaffected by artifacts or missing pixels beyond that padding.

The challenge with this approach is in constructing the $K_j$ approximations; finite support in the image domain makes it impossible to formally also make $K_j$ band-limited, but we need to keep aliasing due to high-frequency power in $K_j$ to a minimum.

### 2.2 Combining resampling and weights

#### 2.2.1 GALSIM-style Fourier-domain accumulation

#### 2.2.2 Analytic single-kernel image-domain resampling and weights

## 3 Extensions for relaxed assumptions

### 3.1 Nonstationary noise

### 3.2 Undersampled input images

### 3.3 Missing pixels

### 3.4 Chromatic PSFs

## A References

## B Acronyms

| Acronym | Description |
|---|---|
| DM | Data Management |
| DMTN | DM Technical Note |

| | |
|---|---|
| FFT | Fast Fourier Transform |
| PSF | Point Spread Function |
| SED | Spectral Energy Distribution |