Draft version January 31, 2022
Typeset using LaTeX **preprint** style in AASTeX631

Regularized Maximum Likelihood Techniques for ALMA

Brianna Zawadzki [1,2,3] Ian Czekala [1,2,4,3] and Ryan A. Loomis [5]

[1] *Department of Astronomy and Astrophysics, 525 Davey Laboratory, The Pennsylvania State University, University Park, PA 16802, USA*

[2] *Center for Exoplanets and Habitable Worlds, 525 Davey Laboratory, The Pennsylvania State University, University Park, PA 16802, USA*

[3] *Institute for Computational & Data Sciences, The Pennsylvania State University, University Park, PA 16802, USA*

[4] *Center for Astrostatistics, 525 Davey Laboratory, The Pennsylvania State University, University Park, PA 16802, USA*

[5] *National Radio Astronomy Observatory, 520 Edgemont Rd., Charlottesville, VA 22903, USA*

Contents

1. EXECUTIVE SUMMARY

The application of Regularized Maximum Likelihood (RML) imaging techniques to sub-mm interferometric observations can achieve higher angular resolution and superior image fidelity compared to traditional imaging methods like CLEAN, as shown in the analysis of the recent Event Horizon Telescope images of M87. While utilizing RML techniques for ALMA data opens the door for higher resolution imaging of both archival and future observations, how to most effectively employ RML for various ALMA measurement sets has previously remained unclear.

Here, we explore the behavior of various prior distributions (maximum entropy, sparsity, total variation, and total squared variation) on ALMA observations in order to understand how each prior impacts the final image and in which imaging cases certain priors work best, both individually and in combination with each other. Deepening our understanding of the interplay between different priors allows us to identify heuristics for different morphological cases, providing a clearer path towards getting started with RML for those who may have otherwise relied solely on CLEAN for their imaging needs. Though we have focused our exploration on protoplanetary disk morphologies, we identify which prior choices are likely to be suitable for certain types of features, such as sharply defined rings or diffuse emission.

Tuning the strengths of each prior is a key part of the RML workflow, as these parameters determine the final outcome of the imaging process. We develop and distill recommendations for incorporating the image validation procedure cross-validation (CV) into the RML workflow in order to obtain a final product with the best possible angular resolution and image fidelity. We detail the general CV process, which involves testing subsets of data in multiple rounds to assess general model performance, including different methods of CV and nuances that accompany image validation for RML.

RML imaging techniques can require a large amount of computational resources, especially for images with a high number of pixels or spectral line data with many velocity channels. This can create a computational burden that must be overcome in order to make RML a practical and accessible method of imaging. We have successfully used the open-source Python package MPoL with GPU acceleration to quickly (on order a few seconds) optimize an image with RML. Harnessing the power of GPUs with MPoL also allows for completion of CV for a set of model parameters in only a few minutes.

Lastly, we compare RML performance to CLEAN performance, providing a direct comparison of RML and CLEAN images for the protoplanetary disk HD 143006. Details of the RML workflow for this specific case are provided, as well as more general recommendations geared towards the community for using RML imaging techniques on ALMA measurement sets.

2. BACKGROUND

The Atacama Large Millimeter/submillimeter Array (ALMA) is capable of observing sources at high angular and spectral resolution. With its 66 12-m antennas and baselines of up to 16 km, ALMA can achieve 20 mas resolution at 230 GHz, placing it at the forefront of submillimeter interferometry. Obtaining high resolution images has enabled progress on a myriad of science goals across a variety of subfields, such as characterizing the dust and gas substructures of protoplanetary disks (e.g. ALMA Partnership et al. 2015; Andrews et al. 2016, 2018; Huang et al. 2018). While high resolution observations from interferometers like ALMA have already furthered our understanding of disks,

galaxies, black holes, and more, there are still many outstanding questions that will not be answered until we can resolve sources to even smaller scales.

Interferometers are composed of a number of individual antennas, with each pair of antennas defining an additional baseline. Because there are a finite number of antennas in the interferometer, every possible baseline length cannot be represented during an observation. This leads to incomplete sampling of spatial frequencies (u, v), regardless of the array configuration. As a result, interferometers incompletely and noisily sample the visibility function of an astronomical source, given by

$$\mathcal{V}(u,v) = \iint I(l,m) \exp\left\{-2\pi i(ul+vm)\right\} \mathrm{d}l \, \mathrm{d}m. \tag{1}$$

Here, $\mathcal{V}(u, v)$ is the visibility function parameterized by spatial frequencies u and v, and $I(l, m)$ is the sky brightness distribution, where $l = \sin(\Delta\alpha \cos\delta)$ and $m = \sin(\Delta\delta)$ for right ascension α and declination δ.

The visibility function $\mathcal{V}$ and the sky brightness distribution I are related by Fourier transform. However, because $\mathcal{V}$ is incompletely sampled, the final image product depends on which assumptions are made about the unsampled spatial frequencies. If all unsampled spatial frequencies are set to zero power, the Fourier transform of the visibilities creates the *dirty image*. The dirty image can be thought of as a convolution of the true sky brightness distribution and the instrument point spread function (PSF), or dirty beam. Because beam sidelobes add image artifacts that are not representative of the true source sky brightness, dirty images require processing to better reconstruct the sky brightness.

Image-plane deconvolution algorithms like CLEAN have been the gold standard in radio astronomy for decades. CLEAN is currently one of the most popular and well-supported imaging methods in the community, and leans heavily on the above idea that the dirty image is the result of the true sky brightness being convolved with the PSF of the instrument. CLEAN works by beginning with the dirty image and iteratively deconvolving beam sidelobes while building up a model representation of the sky brightness (Högbom 1974). The model is built from CLEAN components, usually Dirac δ-functions or Gaussians, which are placed at the location of the brightest pixel in the dirty image. Deconvolution occurs when the CLEAN component is convolved with the dirty beam and subtracted from the dirty image. This process repeats until either a certain number of iterations has been reached, or the dirty image reaches some noise threshold. There are two products at the end of the CLEANing process: the residual image (originally the dirty image, but now contains only residuals after deconvolution) and the CLEAN model (composed of CLEAN components). The CLEAN model is then convolved with the CLEAN beam, which is usually a Gaussian fit to the main lobe of the dirty beam. Finally, the residual image is added to obtain the final CLEANed image.

Although CLEAN has contributed to many advancements in astronomy, it does have limitations. First, while CLEAN does attempt to remove imaging artifacts caused by the PSF sidelobe response, it does so imperfectly. CLEAN components are typically simplistic (e.g. a Gaussian), which may not be suitable for capturing certain morphologies, such as sharp edges or rings. Because CLEAN images are made by adding components that are unlikely to accurately represent a spatially-resolved source, the final image is usually convolved with a final restoring beam. This convolution makes the CLEANed image more visually pleasing, but generally acts as a low pass filter, spatially broadening any high resolution information in the CLEAN model. In other words, convolution with the CLEAN beam imposes a resolution limit on the final image. Other drawbacks include the computation speed,

as CLEANing even a small image can take several hours. Lastly, CLEAN is a nonlinear procedural method of image restoration rather than a true optimization algorithm. There are many user-set algorithm parameters that affect the outcome of the CLEANing process (e.g. stopping criteria), many of which do not have a clear best choice that corresponds with optimal image fidelity.

With modern computing abilities, Regularized Maximum Likelihood (RML) techniques for imaging have become powerful alternatives to more traditional deconvolution procedures like CLEAN. RML techniques have been well-established in the field for decades (e.g. Cornwell & Evans 1985; Narayan & Nityananda 1986), and have been shown to achieve higher angular resolution in sub-mm continuum observations than CLEAN. For example, Pérez et al. (2019) successfully used RML techniques to obtain an image of the protoplanetary disk HD 169142 with higher resolution than the corresponding CASA `tclean` image with Briggs weighting.

At its core, RML is simply an optimization problem; the image is constructed by maximizing some likelihood function defined for some image parameterization. A common way to parameterize the image is by the set of pixel intensities, allowing each pixel to change freely during optimization. The likelihood function encompasses assumptions about the noise properties of the data, as well as any additional information that should be taken into account during imaging. The high degree of flexibility built into the RML imaging process makes it possible to significantly improve image fidelity compared to the dirty image, and because RML techniques require no restoring beam, significantly improve angular resolution. These improvements also hold up to comparisons with CLEAN. Chael et al. (2016) show that for compact sources the normalized root-mean-squared error (NRMSE),

$$\mathrm{NRMSE} = \sqrt{\frac{\sum_{i=1}^{n^2} |I'_i - I_i|^2}{\sum_{i=1}^{n^2} |I_i|^2}}, \tag{2}$$

where I' is the restored image and I is the true image, is minimized at a considerably smaller beam size with RML techniques compared to CLEAN.

However, optimizing the model to obtain the best possible image from the sample visibilities with RML is not trivial, as there are an infinite number of possible sky brightness distributions that correspond to the incompletely-sampled spatial frequencies. The dirty image, generated from the assumption that the unsampled frequencies do not contain any power (i.e. they are set to zero), is just one of many possible maximum likelihood solutions. Regardless of the source morphology and the array configuration, however, this assumption is unlikely to be valid. Including thoughtful choices of priors (via the likelihood function) in the RML workflow allows for substantial improvement of the final image compared to both the dirty image and deconvolution-based procedures like CLEAN.

There have already been many instances of interferometric observations benefitting from RML imaging techniques. A well-known example is the analysis of the Event Horizon Telescope images of M87. Event Horizon Telescope Collaboration et al. (2019) successfully used two independently-developed RML pipelines to obtain high-resolution images of M87. In order to directly compare with CLEAN results, the RML images were restored with beams large enough to reduce the resolutions to that of the CLEAN image. The blurred RML images characterized the ring diameter and asymmetry consistently with the CLEAN image, showing that RML methods can produce higher resolution images at similar image fidelity requirements. RML imaging techniques have also contributed to analyses of ALMA continuum observations of protoplanetary disks. Cárcamo et al. (2018) used RML with entropy-based regularizers on ALMA continuum observations of HL Tau, finding that RML methods

can not only achieve better resolution than CLEAN, but also remove systemic background noise more effectively. Yamaguchi et al. (2020) applied RML imaging techniques with sparsity and total squared variation regularizers to ALMA observations of the protoplanetary disk around HD 142527, finding that RML techniques result in improved image fidelity and higher angular resolution compared to CLEAN images.

This paper is organized as follows. We describe the data used in this study and how it was prepared for RML imaging in section 3. In section 4 we discuss the theory of RML imaging, including the mathematical forms of various regularizers, a technical overview of the RML imaging Python package MPoL, and image validation procedures. We discuss the behavior and attributes of different regularizing terms in section 5, followed by a discussion on characterizing the resolution of RML images and a more thorough examination of image validation procedures in section 6. Finally, we present our recommendations for developing a successful RML workflow for ALMA measurement sets in section 7.

3. DATA

Throughout this study, we used two reference ALMA visibility datasets. The first is a small, mock dataset created from the ALMA logo. Its small size means that it can be easily stored and processed on servers with limited computational means[1]. A full accounting of the processing steps are available on the `mpoldatasets` repository;[2] we briefly summarize them here. The logo was converted to a greyscale image, Fourier transformed, apodized with a Blackman Harris window function (to remove spatial frequencies substantially higher than will be sampled by the target array), and saved as a FITS file. We then used simobserve (CASA version 6.1 McMullin et al. 2007) with the C43-7 reference ALMA configuration from simobserve (`alma.cycle7.7.cfg`), to "observe" the source as it transits zenith for 1 hour under median conditions. Example RML images generated from this dataset can be seen in Figure 2.

The second dataset we used in this study is a real Band 6 ALMA dataset containing the observations of the protoplanetary disk hosted by HD 143006 obtained by the DSHARP survey (Andrews et al. 2018), achieving a resolution of 45 milliarcseconds. We chose this protoplanetary disk because of its public and well-known nature, as well as its potential for structures at small spatial scales, including azimuthal asymmetries. The visibilities were originally calibrated by the DSHARP team following the standardized CASA procedures described in Andrews et al. (2018). The full listing of archival observations can be found in Andrews et al. (2018, Table 3), and the calibrated visibilities can be downloaded from the DSHARP archive[3]. We performed one additional step of calibration beyond that of the DSHARP team. We found that the definition of the visibility weights was not consistent across all of the archival datasets, most likely because the treatment of statistical weights frequently changed in 4.x versions of CASA used to originally calibrate the visibilities. We found empirical weight scalings for each dataset by creating a `tclean` model, subtracting it from the visibilities, and examining the scatter in the visibility residuals compared to the Gaussian envelope expected from the thermal weights. A walkthrough of this rescaling process is described as part of the MPoL documentation[4] and is documented in the `mpoldatasets` repository[5].

[1] Such as those used for continuous integration of code on GitHub, which is one reason why this dataset appears in many of the tutorials that accompany the MPoL documentation

[2] https://github.com/MPoL-dev/mpoldatasets

[3] https://almascience.eso.org/almadata/lp/DSHARP/MSfiles/HD143006_continuum.ms.tgz

[4] https://mpol-dev.github.io/visread/tutorials/rescale_AS209_weights.html

[5] https://github.com/MPoL-dev/mpoldatasets/tree/main/products/HD143006-DSHARP-continuum

4. FORWARD MODELING WITH RML

From a probabilistic viewpoint, RML imaging aims to maximize the posterior distribution of the image, given the dataset and specified priors. Starting with Bayes theorem and acknowledging that the model parameterization will remain fixed[6], we obtain

$$p\left(\mathbf{I} \mid \mathbf{D}\right) \propto p\left(\mathbf{D} \mid \mathbf{I}\right) p(\mathbf{I}) \tag{3}$$

where $p(\mathbf{I} \mid \mathbf{D})$ is the posterior distribution to be maximized, $p(\mathbf{D} \mid \mathbf{I})$ is the likelihood function, and $p(\mathbf{I})$ is the prior distribution.

We make assumptions about the data generating process via the likelihood function. In computing, however, the use of the log likelihood is typically favored. Assuming that the noise is uncorrelated and follows a normal distribution with standard deviation σ, the natural logarithm of the likelihood function is

$$\ln p\left(\mathbf{D} \mid \mathbf{I}\right) = -N \ln(\sqrt{2\pi}\sigma) - \frac{1}{2}\sum_{i}^{N}\left(\frac{\mathbf{V}_i - \mathcal{V}_i}{\sigma}\right)^2. \tag{4}$$

Here, $\mathbf{V}_i$ is measured visibility data at a (u, v) point and $\mathcal{V}_i$ is the predicted value of the model for the same (u, v). Except for the factor of $1/2$, the rightmost term above is simply a χ^2 distribution,

$$\chi^2(\mathbf{D} \mid \mathbf{I}) = \sum_{i}^{N}\left(\frac{\mathbf{V}_i - \mathcal{V}_i}{\sigma}\right)^2, \tag{5}$$

and the log likelihood can be expressed as

$$\ln p\left(\mathbf{D} \mid \mathbf{I}\right) = -\frac{1}{2}\chi^2(\mathbf{D} \mid \mathbf{I}) + C. \tag{6}$$

We can now see that in order to maximize the log likelihood (which, if the priors are flat, will also maximize the posterior), it is necessary to minimize $\chi^2(\mathbf{D} \mid \mathbf{I})$. Rather than maximizing the log likelihood, however, in computing it is more common to minimize the negative log likelihood, given by

$$L_{\rm nll}(\mathbf{I}) = -\ln p\left(\mathbf{D} \mid \mathbf{I}\right) = \frac{1}{2}\chi^2(\mathbf{D} \mid \mathbf{I}). \tag{7}$$

In the machine learning community, it is common to focus on the optimization of some metric that can be described by a *loss function* (e.g. Bishop 2006; Hastie et al. 2009; Murphy 2012; Deisenroth et al. 2020). A loss function is some function which, when minimized, yields optimal parameter values. The negative log likelihood is a reasonable choice of loss function for data with Gaussian uncertainties before considering any regularization. Regularization allows additional information to be considered during optimization via the loss function, lessening the ill-conditioned nature of the problem. While the Bayesian framework involves maximizing the posterior distribution given some additional prior probability distributions, the maximum likelihood perspective involves maximizing the likelihood function by mimimizing the loss function. We use the notation $L_{\rm loss}(\mathbf{I})$ to describe loss

[6] We focus on models that are discretely parameterized by pixel intensities, but there are many other ways to parameterize an image model which we do not discuss here (e.g., parameterization by wavelet coefficients as in Carrillo et al. 2014). Regardless, the chosen model parameterization $p\left(\mathbf{D}\right)$ must remain fixed to obtain the proportional relationship expressed in Equation 3.

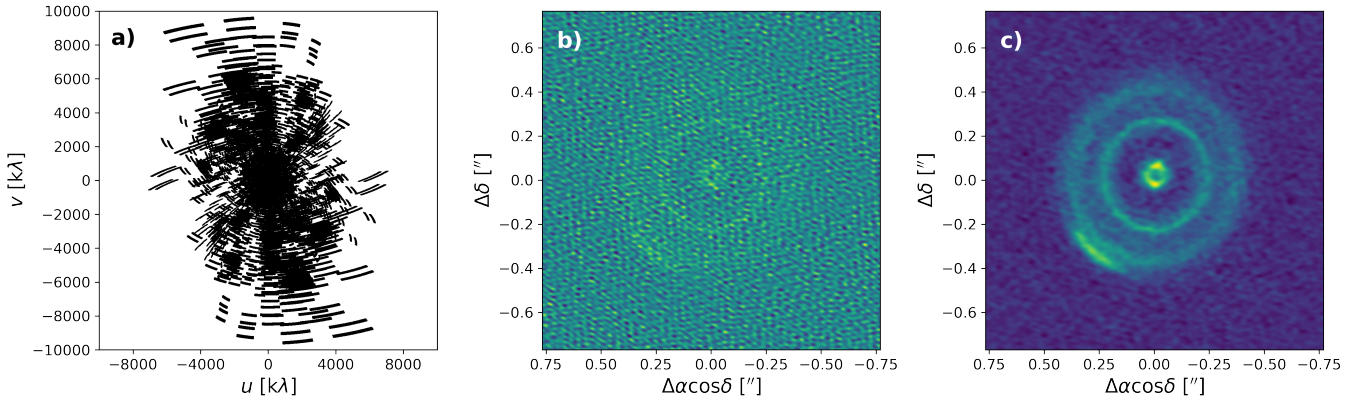

Figure 1. An example of how incomplete (u, v) sampling can impact the appearance of the dirty image. a) The (u, v) sampling of the HD 143006 dataset. The centermost visibilities correspond to the shortest baselines, while the outermost visibilities correspond to the longest. Even though there are long-baseline visibilities present in the dataset, they are mostly sparsely sampled. Gaps in (u, v) space at a variety of baselines create artifacts in the dirty images because the power at these unsampled spatial frequencies is assumed to be zero. b) The dirty image made with MPoL using uniform weighting, which favors resolution over sensitivity. c) The dirty image made with MPoL using Briggs weighting with a robust parameter of 0.0, which creates a more balanced tradeoff between resolution and sensitivity.

terms within the loss function, such as the negative log likelihood or any regularizers being used to better specify the problem.

Though the negative log likelihood can function independently as a loss function, it provides no additional information about what characteristics the final image should have, yielding an unregularized fit. In radio interferometry, minimizing the negative log likelihood of the data alone often results in an undesirable final image. This is due to the incomplete sampling of the visibility function at certain spatial frequencies; if the visibility function has significant power in (u, v) space that is unsampled or only sparsely sampled, minimizing a loss function with no regularization will result in an image that deviates from the true sky brightness distribution. Figure 1 shows the (u, v) sampling of the HD 143006 dataset alongside the dirty images made by averaging visibilities with both uniform and Briggs weighting. The visibility function likely has power at some of the unsampled spatial frequencies. When these unsampled but presumably nonzero powers are set to zero, the resulting dirty images contain artifacts that are not likely to be physical, such as blotchy emission or noisy background.

In cases like these, one can mitigate these effects by incorporating regularization into the imaging workflow. The image can be regularized by incorporating prior knowledge about the source morphology into the model, pushing the final image to adopt certain characteristics during the optimization process. Such constraints can be included by adding regularizing terms to the loss function. For example, a loss function that includes regularization could be of form

$$L(\mathbf{I}) = L_{\text{nll}}(\mathbf{I}) + \lambda_{\text{A}} L_{\text{A}}(\mathbf{I}) + \lambda_{\text{B}} L_{\text{B}}(\mathbf{I}), \tag{8}$$

but more or fewer losses could be included depending on the needs of the specific problem. Each λ value allows the strength of each regularizer to be tuned. Tuning this parameter is important in order to prevent over-fitting the model, as a poorly-weighted regularizing term will result in a final image

that is a poor match to the data. However, a loss function can include terms with a wide variety of functional forms, and whether a λ prefactor is required simply depends on the functional form of each regularizer. Note that these regularizing terms correspond to priors in the Bayesian framework, as they both impose some existing knowledge or expectation about the source on the model.

It is possible to proceed with only a handful of parameters to describe the model. For example, a parametric model for a protoplanetary disk might be defined as a set of annular rings, each further specified by their radius and intensity. However, we may not know enough about the source to place such constraints on the model, and a non-parametric approach can offer more flexibility during the optimization process. Consider an image with $N \times N$ pixels. Each pixel has some intensity I_N such that the image is described by a set of pixel intensities,

$$\mathbf{I} = \{I_1, I_2, \ldots, I_{N^2}\}\,. \tag{9}$$

We can use the intensity of each pixel in the image cube as model parameters, allowing each pixel to vary freely. This introduces a great deal of flexibility into the imaging process, and a loss function with carefully-tuned regularizing terms can harness this flexibility to significantly improve image fidelity compared to a parametric fit.

4.1. *Minimizing the Loss Function*

Minimizing the loss function maximizes the likelihood function, giving a "best fit" image that can change based on what kind of regularization is implemented. There are a wide variety of optimization algorithms that can be used for this minimization problem. We use gradient descent methods, which are iterative processes with several components. First, the gradient of the loss function is computed with respect to model parameters,

$$\nabla L(\mathbf{I}) = \left\{ \frac{\partial L(I_1)}{\partial I_1}, \frac{\partial L(I_2)}{\partial I_2}, \cdots, \frac{\partial L(I_{N^2})}{\partial I_{N^2}} \right\}. \tag{10}$$

Here, the set of model parameters is equivalent to the set of pixel intensities. In order to begin optimization, it is necessary to select an initial set of pixel intensities to be evaluated against the loss function in the first iteration. There are multiple options for setting the initial set of pixel intensities. The simplest starting point is a constant image; if all model pixels are given the same initial intensity, the optimization process starts from a completely blank slate.

However, a faster alternative is initializing the model with some prior knowledge about the true sky brightness. In most cases we already have some initial information about the source: the dirty image. Although the dirty image is not the desired final result, it is the Fourier transform of the sampled visibilities and one maximum likelihood solution to the data. Because of this, it contains useful information about the sky brightness and provides a starting point for the model that is probably closer to the final regularized image than constant pixel values. It is also advantageous to start optimization with the dirty image because the model will likely converge in fewer iterations than if the initial state of the parameters had been uniform (or in any other configuration that is unlikely to represent the true sky brightness, as shown in the third column of Figure 2).

While the dirty image or an image of uniform intensity are reasonable starting points, the initial values of the model parameters could be essentially anything. If the loss function space is convex (i.e. has only a single global minimum), the model will converge to the same result regardless of the

initial state of the parameters. The loss surface is convex for the regularizing terms presented here (e.g. see Akiyama et al. 2017a; Chael et al. 2018; Yamaguchi et al. 2020), so a poor choice of initial pixel intensities comes only at the cost of requiring more iterations to converge on a minimum loss value.

Figure 2 shows how different sets of initial pixel intensities impact the speed of convergence while regularizing a sky brightness projection of the ALMA logo with added noise. We apply entropy, sparsity, and total squared variation regularizers to the loss function. The dirty image converges first, the blank image second, and the custom image last. We use a custom image of a dog, intentionally selecting a set of pixel intensities with no similarity to the true image. Though the custom set of initial pixel intensities takes significantly longer to converge, it ultimately does converge on the same result as the initial dirty and uniform images, showing that the final result is not sensitive to the initial state of the model. Figure 2 also shows how the model image is updated during optimization: after each iteration, the gradient of the image is added to the model parameters, creating a new model image. This process repeats until the loss function converges on a minimum, and the gradient is zero or approximately zero.

One important consideration with the gradient descent method is step size. Steps that are too large could overshoot the minimum, causing the algorithm to diverge. The smaller the step size, the more iterations will be required for the loss function to converge on a final value, meaning that steps that are too small can quickly become computationally expensive. It is worthwhile to check that the loss function has actually converged on a value; an image that has not been fully optimized can be misleading as it is not actually the maximum likelihood solution. For example, in Figure 2 the model images at 50 and 150 iterations look quite similar. However, the magnitude of the gradient image is reduced by several orders of magnitude at 150 iterations. We can also see that the the loss function has not yet been minimized at 50 iterations, especially when the model was initialized with a blank or custom image. Though it may be tempting to run fewer iterations in the interest of computational speed, one should take care to ensure that the loss function has fully converged.

4.2. *Regularizers*

The actual changes to the model image throughout the optimization process are guided by the choice of regularizers included in the loss function. A regularizer can be thought of as a stabilizing force on the image parameters, for example, penalizing pixels that poorly match the expected sky brightness. Returning to a Bayesian viewpoint, including a regularizer is analogous to imposing a non-uniform prior on the data (Sivia & Skilling 2006). Implementing regularizers in the imaging process effectively allows us to make assumptions about unsampled and noisily sampled frequencies based on our prior knowledge of the source. In practice, one may need to use a combination of several regularizers to obtain a final image that best represents the true sky brightness.

Different types of regularizers vary in their implementations and subsequent effects on the image. Some regularizers are implemented by construction, meaning there is some part of the imaging workflow that inherently imposes the regularizer without needing to tune its strength or other hyperparameters. Other regularizers are imposed directly on image pixels. These pixel-based regularizers typically have an associated strength factor that determines how great the overall effect on the final image will be. In addition, they may require tuning internal parameters which affect the pixel-by-pixel behavior of the regularizer. Lastly, it is possible to regularize not over the image itself, but over some quantity derived from image properties. These regularizers affect pixel values indirectly

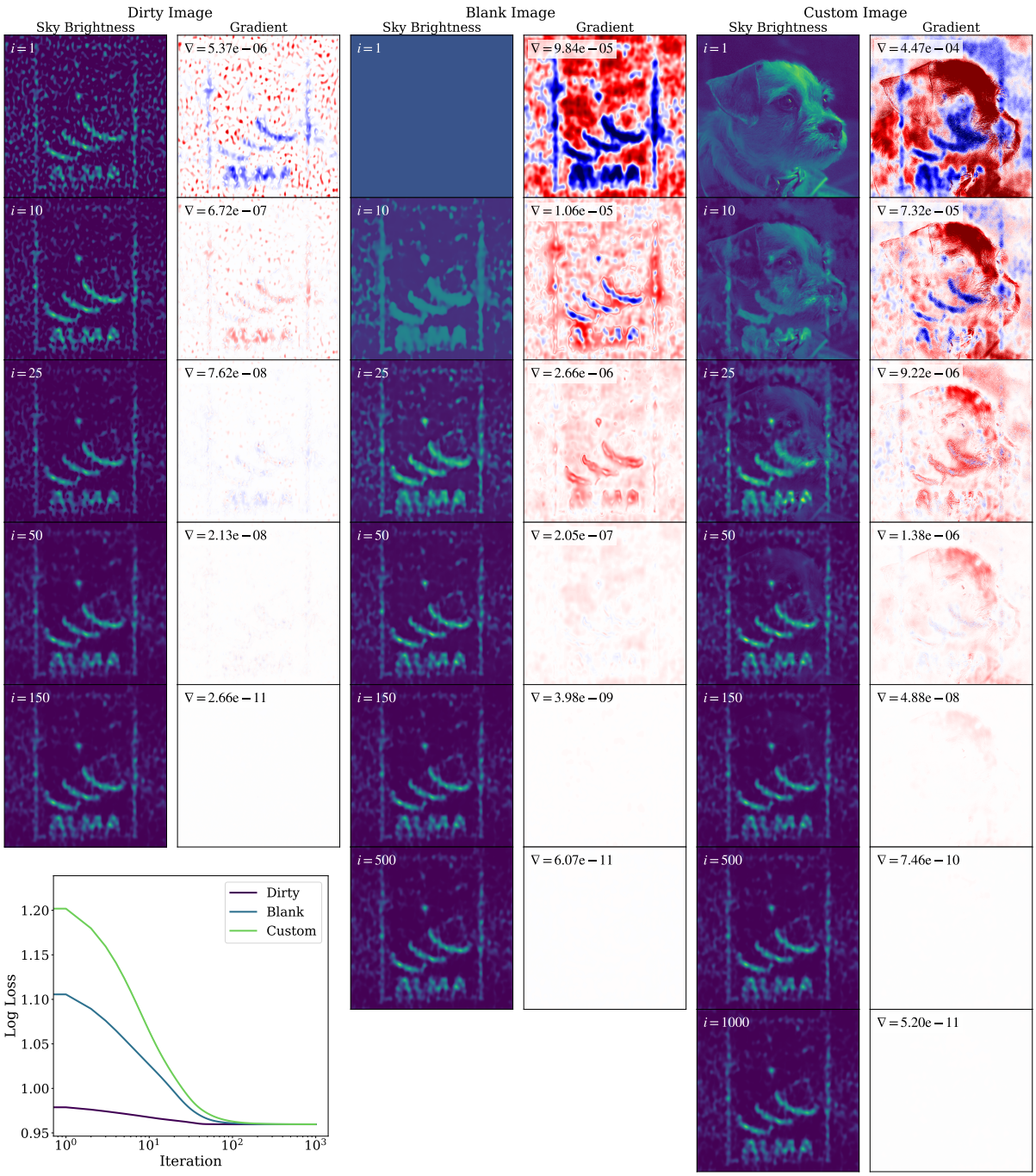

Figure 2. Visualizing the regularization process using gradient descent optimization to minimize the loss function for different initial pixel values. A combination of entropy, sparsity, and total squared variation regularizing terms were used on a mock dataset made from the ALMA logo with added noise. Each row shows the state of the model and gradient image at a different number of iterations during optimization. The leftmost columns show the sky brightness and gradient images of a model initialized with the dirty image, the middle columns show the same for a model initialized with a blank image, and the rightmost columns show the same for a model initialized with a custom image (in this case, an image of a dog). From left to right, this can be read as the best to worst guess for the true sky brightness distribution. In each gradient image, positive (negative) values are shown in red (blue), and ∇ denotes the magnitude of the gradient vector (i.e. the gradient value of each pixel added in quadrature). All sky brightness and gradient images are plotted on the same color scale. A good set of initial pixel values like the dirty image quickly converges to the final image, while a poor guess is more computationally expensive to achieve the same result.

based on how changes to the model sky brightness impact the derived quantity. Here we discuss the functional form and motivation for each regularizer we tested.

4.2.1. *Image Positivity*

Image positivity requires that pixel values be non-negative. From a physical standpoint it makes sense to assume that the intensity value of a given pixel must be positive, as the flux of the observed source must also be positive (or zero, if there is no flux). Though we do not have intensity information at all spatial frequencies, we can use image positivity regularization to impose a restriction on the final image that each pixel must have a non-negative intensity value.

Because positivity can be enforced implicitly through the definition of the model image, image positivity is an example of a regularizer implemented by construction. Despite the apparent simplicity of requiring positive pixel values, there are several possible ways to impose image positivity. The rectified linear unit (ReLU) activation function,

$$f_{\mathrm{ReLU}}(x) = \max(0, x), \tag{11}$$

returns an unchanged input value if the input is positive, and returns zero otherwise. However, interferometers have a finite number of baselines and do not completely sample the visibility function of the source. Considering that the visibility function and sky brightness distribution are related by Fourier transform (Equation 1), this incompleteness causes negative-valued pixels in the sky brightness image. This is true even for perfectly-measured visibilities, though noise considerations further complicate the issue.

Using a modified version of the ReLU function can ensure that each sky brightness intensity pixel is strictly positive while still retaining some information about the relative brightness of each pixel. One such example is Softplus, which has the functional form

$$f_{\mathrm{Softplus}}(x) = \frac{1}{\beta} * \log(1 + \exp(\beta * x)), \tag{12}$$

where β is a sharpness parameter. This is a smooth approximation to the ReLU function, but instead of setting all negative values to zero, Softplus allows negative input values to have a positive nonzero output without drastically changing large positive input values. In other words, the Softplus mapping retains information about the image from the sampled visibilities while eliminating the problem of negative-valued pixel intensities caused by the incomplete visibility sampling.

4.2.2. *Maximum Entropy*

In general, maximum entropy regularization promotes images with similar pixel values to a given set of reference pixels. Maximum entropy regularization is well-established in radio interferometric imaging, and is known to achieve higher resolutions than corresponding CLEAN images (Cornwell & Evans 1985; Narayan & Nityananda 1986). Following the definition in Event Horizon Telescope Collaboration et al. (2019), the maximum entropy loss is formulated as

$$L = \frac{1}{\sum_i I_i} \sum_i I_i \ln \frac{I_i}{p_i}, \tag{13}$$

where p_i is a reference pixel value against which other pixels are compared. These reference pixels could be as simple as a "blank" image of uniform intensity (e.g. Cárcamo et al. 2018), or they could

take additional knowledge about the source into account. For example, Event Horizon Telescope Collaboration et al. (2019) used circular Gaussian images for the set of p_i.

The maximum entropy loss is a pixel-based regularizer, as it is applied to each model image pixel before the final loss value can be calculated. The strength of the regularizer should be adjusted depending on the specific imaging case, so when maximum entropy regularization is incorporated into a loss function, a λ prefactor should be included and tuned as needed.

4.2.3. *Sparsity*

Sparsity regularization uses the L_1 norm to reduce the impact of unneeded pixels, promoting a final image that is a sparse collection of nonzero pixels. Originally stemming from the least absolute shrinkage and selection operator (lasso, see Tibshirani (1996)), sparsity is a pixel-based regularizer that has successfully been applied to radio interferometric imaging to achieve high-resolution images around black holes and protoplanetary disks (e.g Honma et al. 2014; Akiyama et al. 2017a; Kuramochi et al. 2018; Event Horizon Telescope Collaboration et al. 2019; Yamaguchi et al. 2020).

The sparsity loss is formulated as

$$L = \sum_i |I_i| \,. \tag{14}$$

Sparsity regularization reduces the impact of unneeded pixels (i.e., promoting a final image that is a sparse collection of nonzero pixels), making it a useful regularizer when the true sky brightness of a source is likely to be sparse.

Because the sparsity regularizer encourages mostly blank images, faint or diffuse emission may be removed during the optimization process. However, blank regions need not be contiguous, so including a sparsity term will not necessarily regularize out interesting features like disks with rings or other substructures. Regardless of the source morphology, sparsity regularization may not be as effective if the angular pixel size is smaller than the angular area of the emission. However, even if the source is unlikely to be sparse in the image domain (e.g. extended sources like galaxies), sparse regularization has been shown to successfully reconstruct these images by transforming the data such that it is sparse in some other domain (Li et al. 2011; Carrillo et al. 2012). When regularizing with sparsity, a λ prefactor should be included and tuned to adjust the overall strength of the sparsity term.

4.2.4. *Total Variation*

Total variation (TV) regularization promotes images with sharp edges at areas with significant changes in intensity, and relatively smooth areas in between. As a result, an image made with a strong TV term is likely to have a sparse intensity gradient, as the sharp edges are the main contributing components. In other words, the TV regularizer promotes similarity (or smoothness) between adjacent pixels unless the difference is significant, in which case a change in intensity is needed. TV regularization has been used with success on its own and in combination with other regularizers for astronomical interferometric imaging (e.g. Wiaux et al. 2010; Akiyama et al. 2017b,a).

The TV loss is defined by (Rudin et al. 1992) as

$$L = \sum_{l,m,v} \sqrt{\left(I_{l+1,m,v} - I_{l,m,v}\right)^2 + \left(I_{l,m+1,v} - I_{l,m,v}\right)^2 + \epsilon}. \tag{15}$$

The image cube has dimensions where l corresponds to right ascension, m corresponds to declination, and v corresponds to the velocity channel in the cube. The ϵ term is an optional softening parameter which determines how pixel-to-pixel variations within the image slice will be penalized. If adjacent pixels vary more than ϵ the total loss will greatly increase, so TV regularization favors minimal variation between adjacent pixels. The TV regularizer is another type of pixel-based regularizer, requiring a λ prefactor to adjust the overall strength of the regularizer. While ϵ determines how strongly the TV regularizer should penalize pixel-to-pixel variation internally, λ determines how much the entire TV regularizing term should be weighted in the loss function.

4.2.5. *Total Squared Variation*

The total squared variation (TSV) regularizer is a variant of the TV regularizer, still summing the brightness differences between adjacent pixels. However, by not taking the square root of the differences, the TSV prior results in images with smoother edges (Kuramochi et al. 2018). The TSV regularizer,

$$L = \sum_{l,m,v} \left(I_{l+1,m,v} - I_{l,m,v}\right)^2 + \left(I_{l,m+1,v} - I_{l,m,v}\right)^2, \tag{16}$$

is functionally similar to the TV prior, except the expression inside of the summation has been squared.

For astronomical targets that with diffuse features, TSV prior may outperform the TV prior because the edge-smoothing effects better match the expected morphology in the image (e.g. Kuramochi et al. 2018; Chael et al. 2018). As with other pixel-based regularizers, a λ prefactor should be included with the TSV regularizer in order to control the intensity of the TSV regularization.

4.3. *Cross-Validation*

One way to determine whether the regularization (whether it be the strength of the λ prefactors or the functional form of the regularizer itself) is appropriately tuned is by using cross-validation (CV). CV aims to find optimal parameter values by determining how consistently the model performs given variations in the data set, working on the concepts of *training data* and *testing data*. Training data is used to find the model which minimizes the loss function, undergoing the normal optimization process to find the best-fit image given a specified loss function. Testing data is used for comparison against the model optimized with the training data. If some parameter space (such as a set of spatial frequencies) is not covered by the training data, but is covered by the testing data, then comparing the trained model to the testing data effectively measures the predictive power of the model within that parameter space. In other words, testing data allows us to see how well the model predicts new data, and better models correspond to more similarity with the testing data.

In principle, one would like to have enough data to train the model (i.e. run the optimization process with the selected regularizers) and then use a completely separate set of data to test the model. Unfortunately, data is often too limited for this approach; using enough data to train the model typically leaves only a small amount for testing, resulting in a noisy estimate of the predictive performance of the model (Bishop 2006). CV circumvents this limitation by partitioning all the measured visibility data $\mathbf{V}$ into subsets such that $\mathbf{V}_{\text{subset}}(\{u,v\}) \subset \mathbf{V}$, fitting the model on one or more subsets, and testing the model on the remaining subsets. One popular method is K-fold CV, which performs this process in multiple rounds, rotating which subsets are used for testing (e.g.

Akiyama et al. 2017a,b; Yamaguchi et al. 2020). In this case, data are partitioned into K subsets.

$$\mathbf{V} \begin{cases} \mathbf{V}_0(\{u,v\}) \\ \mathbf{V}_1(\{u,v\}) \\ \vdots \\ \mathbf{V}_K(\{u,v\}) \end{cases} \tag{17}$$

After partitioning, $K-1$ subsets are combined to form the training data $\mathbf{V}_{\rm train}$ and the remaining subset is used for testing. The model is optimized using $\mathbf{V}_{\rm train}$ and any model specifications like regularizers. Then, the trained model visibilities are compared against the subset of visibilities that were withheld from the training data. Because

$$p\left(\mathbf{V}_{\rm test} \mid \mathbf{V}_{\rm train}\right) \propto \exp\left[-\frac{1}{2}\chi^2(\mathbf{V}_{\rm test} \mid \mathbf{V}_{\rm train})\right], \tag{18}$$

a smaller χ^2 value corresponds to a better match between the trained visibilities and the testing data.

This process is repeated K times such that each subset functions as the testing data once. We obtain a final CV score by summing the χ^2 values calculated for each of the K CV rounds. A low CV score indicates that the model is well-fitted to the training data. In the case of poorly chosen or poorly tuned regularizers, the model may not fit the training data and can result in a set of visibilities that strongly deviates from the testing visibilities. This large deviation would cause a large CV score, an indicator that the model needs revision.

There are many different ways in which one could partition data for K-fold CV. We explore CV using two methods of partitioning: uniform and dartboard. Uniform partitioning utilizes K subsets that are composed of randomly-selected visibility grid cells. Each grid cell is only used in one subset, so grid cells are randomly drawn without replacement. CV with uniform partitioning tests how the model responds to new data in comparable u-v space to the training data.

Dartboard partitioning splits visibilities along polar grid lines, creating azimuthal and radial bins consisting of many grid cells. Each of the K subsets consists of randomly-drawn dartboard cells without replacement. Figure 3 shows an example of dartboard partitioning. CV with dartboard partitioning shows how the model handles extrapolation to unsampled u-v space. Dartboard partitioning aims to simulate the irregular u-v sampling obtained from different ALMA execution blocks, across slightly different array configurations.

4.4. *The MPoL Package*

Million Points of Light (MPoL)[7] is an open-source Python package designed for RML imaging. As part of the imaging workflow, MPoL provides functionalities for gridding loose visibilities, visualizing dirty images, building custom image models, optimizing the data with respect to user-selected posteriors, and more. MPoL is built on PyTorch (Paszke et al. 2019), an open-source machine learning package that uses tensors optimized for use with CPUs or GPUs. PyTorch provides the ability to calculate gradients on tensors — a key part of the RML imaging workflow with MPoL.

Optimizing images can quickly become computationally expensive, especially if the dataset is an image cube with a large number of pixels and many channels. For this reason, MPoL takes advantage of the power of GPUs, which can greatly reduce computation time compared to CPUs.

[7] https://mpol-dev.github.io/MPoL/

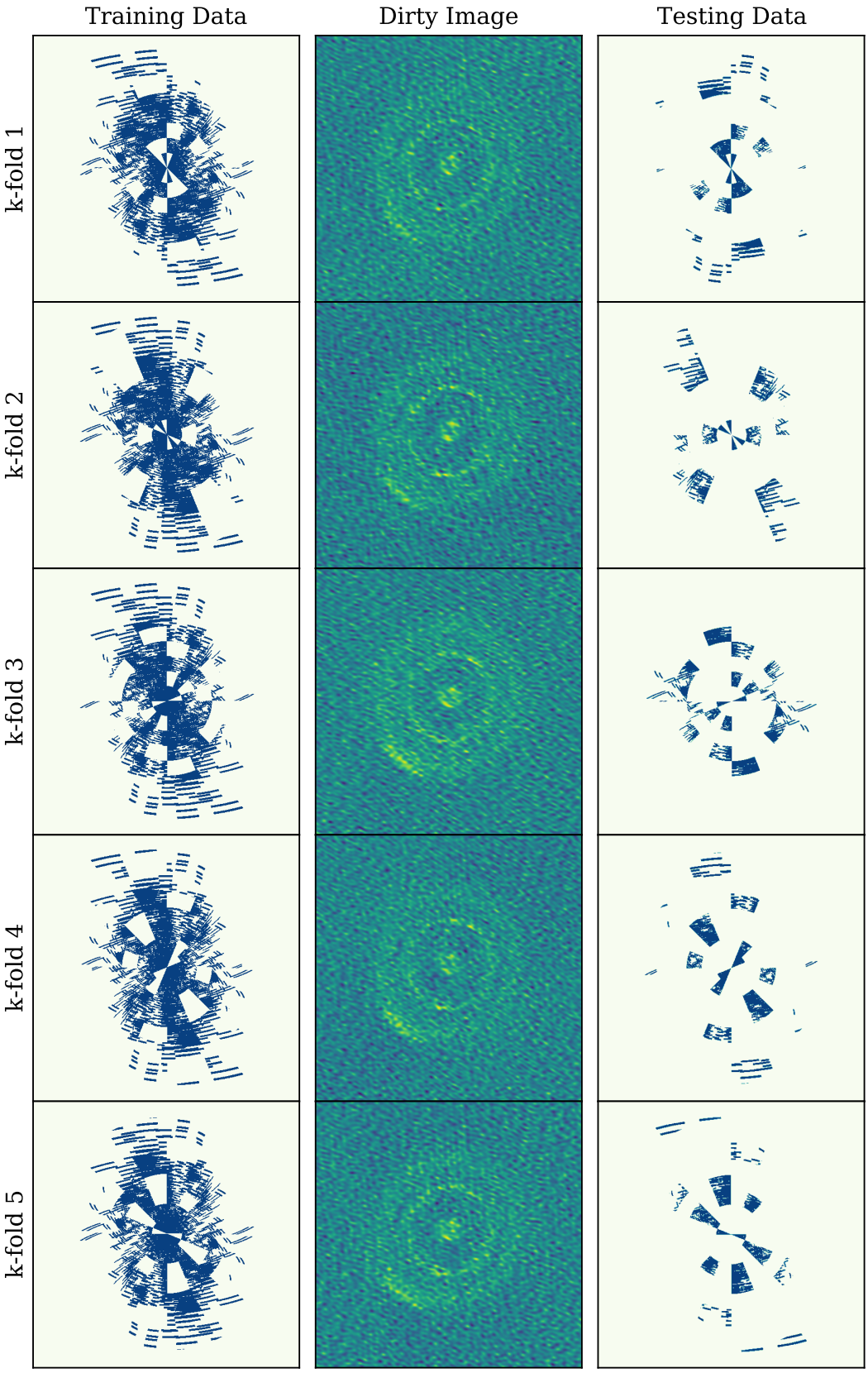

Figure 3. Visualizing the data partitioning for K-fold CV ($K = 5$) of HD 143006 using the dartboard scheme. The left column shows 4 subsets combined and used to fit the model, while the middle column shows the dirty image generated from those 4 subsets. The right column shows the withheld subset of data used for validating the model. Each of the K rows shows a different subset of data used as the testing data.

5. RESULTS

We explored the effects of entropy, sparsity, TV, and TSV regularizers on the HD 143006 data. Figure 4 shows the effect of each of these regularizers at different strengths (set with the λ prefactor on each term). Figure 5 shows the residuals of each optimized image for each of the panels shown in Figure 4. An image of HD 143006 made by combining multiple regularizing terms is shown in Figure 6, along with a comparison to the CLEAN image.

5.1. *Maximum Entropy*

Maximum entropy regularization promotes similarity to a set of reference pixels. Here, we use a uniform set of reference pixels. In other words, we do not directly impose assumptions about the morphology of HD 143006 on the optimization process. Maximum entropy is well-known for its ability to achieve high-resolution images (Cornwell & Evans 1985; Narayan & Nityananda 1986). The first column of Figure 4 shows the effect of different λ prefactors for maximum entropy regularization of HD 143006. Even at high λ values, maximum entropy regularization can retain high-resolution features in the image. However, promoting similarity to an image of uniform intensity means that at excessive λ values, the image tends to a final image that appears "faded," making emission appear fainter across the entire source. The bottom-left panel of Figure 4 shows an example of such an image.

One indication of an underfitting with maximum entropy is a final image that appears faint or slightly blurred. Another way to check for underfitting is by examining the residual image. In the bottom-left panel of Figure 5, ringed structure is evident in the residual image created by maximum entropy regularization with $\lambda = 5$. In some cases, these effects may be mitigated by using a non-uniform set of reference pixels, such as a circular Gaussian (e.g. Event Horizon Telescope Collaboration et al. 2019) or a uniform ring. This should be done with caution, as maximum entropy regularization will tend towards similarity with the reference image, and the reference image may not capture enough characteristics of the true source. This could have unintended consequences such as regularizing out small-scale structures or unexpected asymmetries that are present in the data but not in the set of reference pixels.

5.2. *Sparsity*

Sparsity regularization promotes mostly blank images, with only the most impactful pixels remaining in the final image. Only a λ prefactor determines how strongly sparsity should be imposed during optimization. Column 2 of Figure 4 shows images of HD 143006 with 5 different sparsity λ values. A small λ can effectively remove noisy background pixels in the image without changing much, if anything, about the source emission.

Sparsity regularization alone does not introduce any sort of "smoothing" effects that may be desirable for a resolved source; the final image will ultimately be a sparse collection of nonzero pixels, which can make it difficult or impossible to identify small-scale structures within the image. In the case of HD 143006, sparsity regulariztion is most effectively used in combination with other regularizing terms.

An underfitted sparsity term can have a significant negative impact on the optimization process, creating a final image that is not representative of the entire source. In the case of HD 143006, sparsity regularization with a high λ value removed some of the more diffuse emission, and in extreme cases removed some rings entirely. In Figure 4, the bottom panel of column 2 of shows the result of imaging

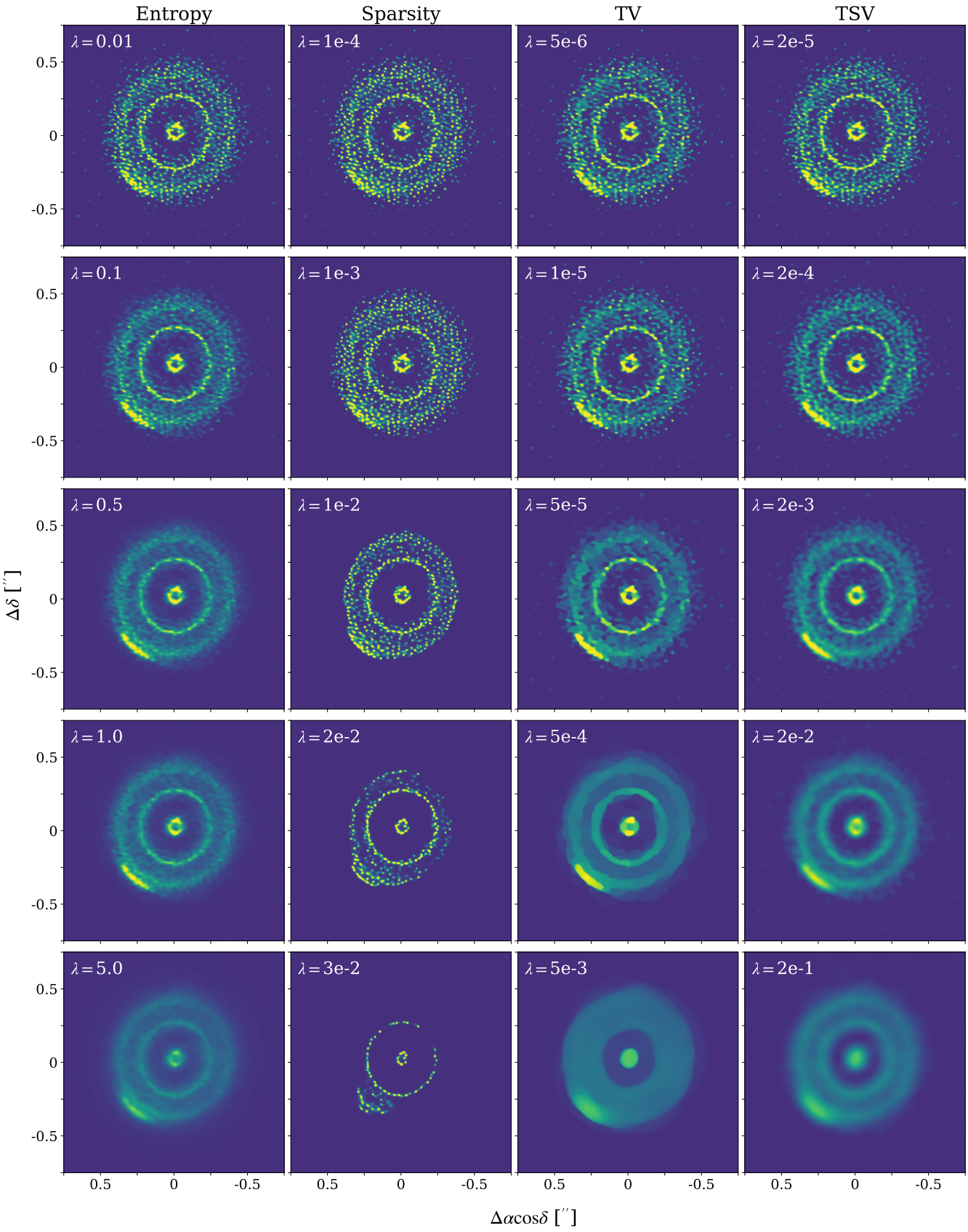

Figure 4. The effect of individual regularizers on images of HD 143006. From left to right, each column shows entropy, sparsity, total variation, and total squared variation regularizers at varying strengths. The top row uses the least regularization (i.e. a small λ prefactor on the regularizing term) and the bottom shows extremely high regularization. The optimal λ values can be found using CV methods.

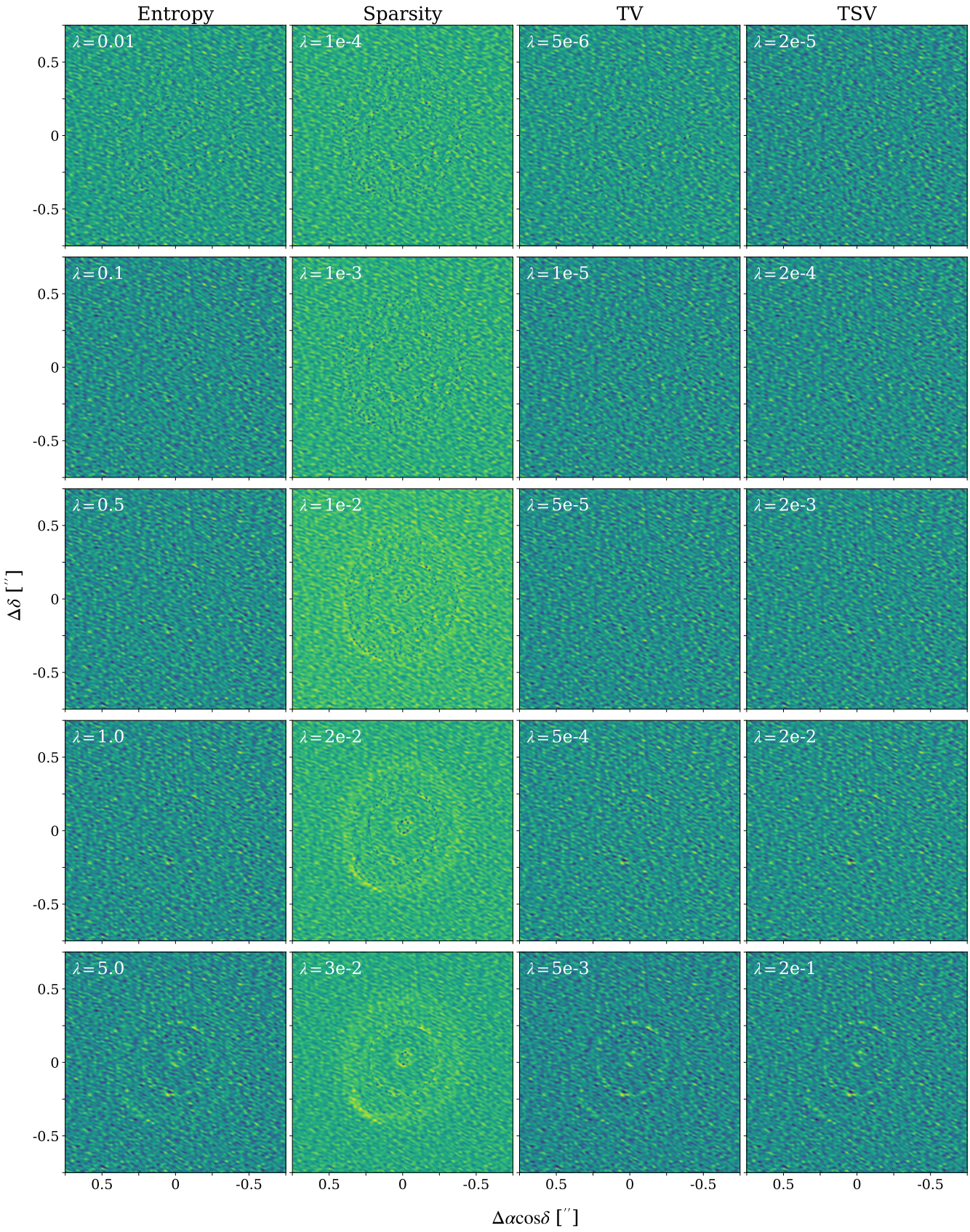

Figure 5. Residuals for the images presented in Figure 4. Regularizer strengths are lowest in the top row, and highest in the bottom row. Images generated with strong regularizing terms show structure in the residual images, indicating that the model is underfitting the data.

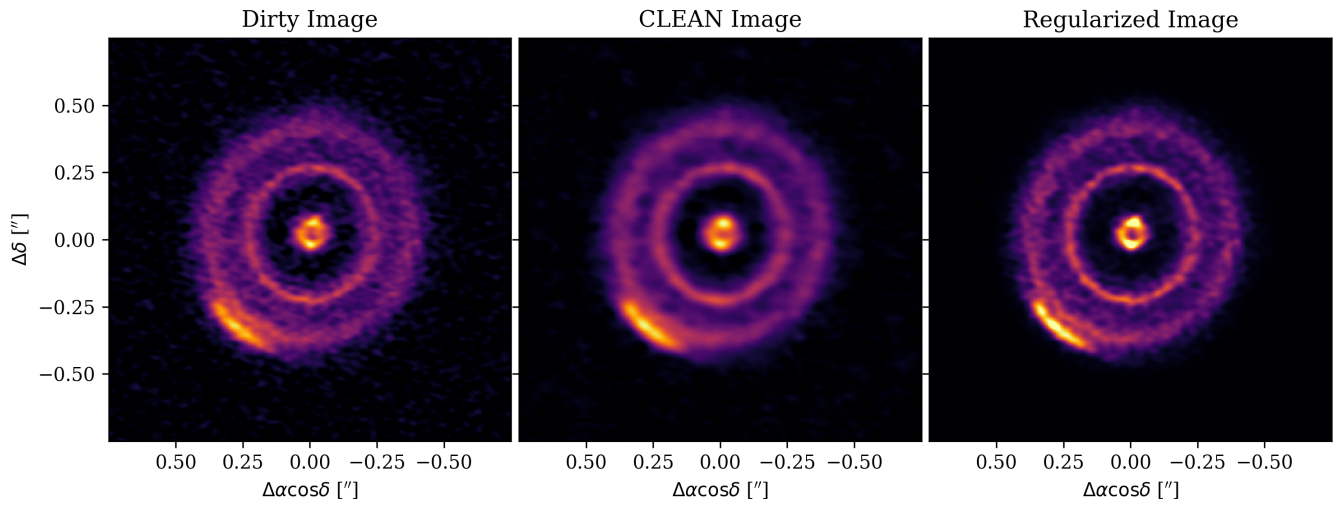

Figure 6. Dirty image (left), tclean (center), and RML (right) images of HD 143006. The loss function for the RML image included maximum entropy ($\lambda = 0.5$), sparsity ($\lambda = 10^{-3}$), and TSV ($\lambda = 2 \times 10^{-3}$) terms. In this case, maximum entropy contributed high resolution features to the final image, sparsity removed background noise, and TSV helped better define ring structure in the disk.

HD 143006 with a highly underfitted sparsity term. Here, the outer ring has been regularized away, but the bright azimuthal asymmetry that normally coincides with the outer ring is still present, completely misrepresenting the morphology of the source.

This sort of underfitting is very evident in the residual image (column 2, Figure 5). Because only the most prominent features remain in the final sky brightness image, any diffuse or generally lower-intensity emission will instead be apparent in the residuals. In the sky brightness image of a resolved source, things to look out for that may indicate underfitting of the sparsity term include features that appear 'incomplete' such as having partial rings, an unexpected bright standalone feature, or unexpectedly sharp changes in intensity.

5.3. *Total Variation*

TV regularization promotes sharp edges between areas of different intensities, with smoothness in areas of similar intensity. Column 3 of Figure 4 shows the effect of different λ prefactors for TV regularization of HD 143006. Because TV promotes similarity between adjacent pixels unless there is a large change in intensity, TV regularization can result in a final image composed of many nearly uniform cells, each containing several pixels. This can create an optical illusion where the image appears to have larger pixels than the true pixel size (e.g. see the third column and third row in Figure 4, where λ=5e-5).

This effect changes as the λ value increases, with final images being reminiscent of a watercolor painting or a photo that has been posterized. Because TV regularization does not favor gradual changes in intensity, instead preferring sharp changes, a smooth change in source intensity is likely to become a set of sharply defined layers in the final image. In the case of extreme overregularization, this can remove most detail from the image, resulting in a final image that appears blotchy or smeared. However, in sources with ringed emission like HD 143006, the smearing is mostly azimuthal rather than radial, retaining the large scale ring structure of the source in the final image while losing or minimizing finer details like gaps or azimuthal asymmetries.

For these reasons, TV regularization may be a poor choice if the source is likely to have small scale features or gradual changes in intensity, as many astronomical sources do. Tuning the λ prefactor for TV regularization is finicky compared to other regularizers, as increasing λ can quickly take the final image from essentially unregularized to the large pixel illusion to completely smeared emission. Though underfitting can be evident due to the presence of structure in residual images (see the bottom panel of column 3, Figure 5), it may not be evident until well after morphological details have already been regularized out of the final image.

5.4. *Total Squared Variation*

Like TV regularization, TSV regularization promotes sharp edges between areas of different intensities. However, the TSV regularizer is less rigid with this condition, allowing for larger differences between adjacent pixels. This makes TSV a strong performer for sources with clearly defined but not perfectly sharp features, such as ringed emission. The rightmost column of Figure 4 shows images of HD 143006 with 5 different TSV λ values. Well-tuned TSV regularization performs comparably to maximum entropy regularization, retaining high-resolution features in the final image.

The primary sign of underfitting with TSV is a blurred image. If the final sky brightness image after regularizing with TSV appears to have no sharp features at all as if it had been put through a low pass filter, it is likely overregularized. This is also evident in the residual images (rightmost column, Figure 5), where sharp structures will appear if they have been regularized out of the sky brightness image.

6. DISCUSSION

6.1. *Subtleties of Cross-Validation*

Cross-validation (CV) can be a useful tool for determining the optimal values of the regularization parameters to maximize image fidelity. The use of CV methods on interferometric data is an active area of research, and there are many aspects which have not yet been fully explored. First, there are both exhaustive and non-exhaustive CV methods; exhaustive methods use all possible ways to partition data, while non-exhaustive methods use only a subset of possible partitions. An example of an exhaustive method is leave-one-out CV (LOOVC), which takes all but one data point as the training set and uses the remaining data to test. These methods can be extremely computationally expensive, especially in the case of interferometric data containing millions of visibilities. In addition, LOOCV performs poorly compared to other methods of CV (Breiman & Spector 1992).

Non-exhaustive CV methods like K-fold CV greatly reduce the total computational burden, and are thus a more practical CV method for interferometric data. K-fold CV requires a choice of K that balances bias and variance, with a high K yielding a low bias, high variance estimation and a low K yielding a high bias, low variance estimation (Hastie et al. 2009). Studies in statistics and informatics have consistently found $K = 10$ to provide the best bias-variance trade-off (e.g. Breiman & Spector 1992; Kohavi 1995; Molinaro et al. 2005). While no studies have specifically examined the optimal K for K-fold CV of interferometric data, the standard $K = 10$ has been used with success for such applications (Akiyama et al. 2017b,a; Yamaguchi et al. 2020).

Aside from the choice of K, one must also decide how to partition the data set. Uniform partitioning, which effectively examines how the model fits with new data in comparable u-v space, has been used in previous studies of RML for interferometry (e.g. Akiyama et al. 2017b,a; Yamaguchi et al. 2020). We explore dartboard partitioning (see Figure 3) to approximate how the model might fit with data

Table 1. An example of how a lack of convergence during CV can yield a misleading final CV score. Here, we compare CV scores for the same set of model parameters, changing only the number of iterations used to train the model. After 1000 iterations, K-fold 3 has not converged, yielding a χ^2 value of 45.15 and greatly driving up the total CV score. After 3000 iterations, the χ^2 value of K-fold 3 is more in line with the other 9 K-folds. This is caused by the training dataset for K-fold 3 not including the lowest spatial frequency visibility cells; only the testing dataset contains these values. For this example we use a dartboard data partitioning scheme on the HD 143006 data, and sparsity ($\lambda = 10^{-3}$) and TSV ($\lambda = 8 \times 10^{-4}$) regularization with a learning rate of 0.5.

			Contribution to total CV score per K-fold									
iterations	time (s)	CV score	1	2	3	4	5	6	7	8	9	10
1000	123.5	62.50	1.88	1.67	45.15	3.06	2.49	1.30	1.18	1.20	2.02	2.55
3000	370.5	14.19	1.31	1.27	1.99	1.68	1.50	1.17	1.15	1.16	1.42	1.55

obtained from a variety of array configurations. However, one need not be limited to these two partitioning schemes, and future work may determine that another method of partitioning visibility data for CV works better.

Regardless of the choice of K and data partitioning scheme, it is vital to ensure that the model has fully converged for each training set. We find that the number of iterations needed to optimize the model during K-fold CV is often greater than the number of iterations needed to optimize the model with the full dataset, as training on fewer data points can require more iterations before the loss function is minimized. This is especially true for the K-fold that does not contain the visibilities at the lowest spatial frequencies (i.e. (u, v) close to zero), as the omission of this data can cause a slow initial decline of the total loss. Because the final CV score is the sum of the χ^2 for all K-folds, a K-fold that has not fully converged can result in a spuriously high CV score. We recommend inspecting each K-fold for convergence, as well as inspecting the final χ^2 value for each K-fold. An unusually high χ^2 value for a single K-fold compared to the other $K - 1$ K-folds can indicate a lack of convergence (see Table 1 for an example). This usually coincides with the K-fold for which the training set lacks data at the lowest spatial frequencies. If each K-fold does not reach convergence, the final CV score is invalid and the entire CV process must be repeated with enough iterations to ensure full convergence.

While CV scores can be used to find optimal regularization parameters, care should be taken when comparing CV scores. First, as mentioned above, any CV score obtained from an incomplete optimization loop (i.e. one or more K-folds do not reach convergence) cannot be used. CV scores may only be compared across the same dataset, model specification, and CV setup. The CV setup includes the value of K and the choice and implementation of partitioning scheme. For example, a CV workflow could be set up with $K = 5$ and dartboard partitioning with 12 radial bins and 16 azimuthal bins, as shown in Figure 3.

It is possible to compare CV scores when using different regularizers, as long as the regularizer has a tuneable prefactor (e.g. λ values) that can be set to zero. For example, one could compare the CV score for a model with only an entropy regularizer to that of a model with only a TV regularizer, because this is effectively comparing different ways to tune the λ prefactors, including $\lambda_{\rm entropy} = 0$ and $\lambda_{\rm TV} = 0$. In addition to these edge cases which "turn off" certain regularizers, CV scores can be compared for any values of $\lambda_{\rm entropy}$ and $\lambda_{\rm TV}$, as long as the dataset and CV setup remain consistent.

6.2. *Determining Image Resolution with RML*

Imaging with CLEAN typically involves building up a model of CLEAN components and then convolving that model with the CLEAN beam. A CLEAN component may be as simple as a Dirac δ-function, which is useful for identifying a location of peak flux but likely to be a poor representation of the spatially resolved source. Beam convolution effectively spreads these components over the size of the beam, making the image more representative of the true source morphology at the cost of resolution. Because the size of the CLEAN beam is a known quantity, characterizing the resolution of the final image is relatively straightforward.

RML images, on the other hand, are not generated from a set of simplistic components and thus do not inherently require beam convolution as part of the imaging process. The most obvious benefit to this is that loss of resolution is not injected into the imaging workflow. However, the lack of restoring beam does make characterizing the final image resolution a bit more ambiguous. Chael et al. (2016) find that restoring beams can still be useful for RML methods, as false high-frequency features can sometimes be injected into the image. While a restoring beam can fix this issue, using too large of a beam could remove real features in the image.

The optimal restoring beam size can be computed given knowledge of the true source; as in Chael et al. (2016), the beam size that minimizes the NRMSE (Equation 2) can be used to maintain the highest degree of superresolution without retaining the spurious high-frequency features. Given that we usually lack the true image, this approach will not always be possible. Even though the optimal restoring beam size cannot always be constrained, image fidelity is only mildly worsened by selecting a beam size smaller than the optimal value. As a result, the use of a restoring beam in an RML imaging workflow can only result in modest fidelity gains in the best case, and can result in significant loss of resolution in the worst case.

7. CONCLUSION

We have developed a GPU-accelerated RML imaging workflow designed for ALMA measurement sets. The RML workflow enables imaging with better resolution and image fidelity than image-plane deconvolution algorithms like CLEAN. We explored how maximum entropy, sparsity, TV, and TSV regularizing terms can be incorporated into the imaging process, and how each of these regularizers affects the final RML image. We also described how to use K-fold CV to validate the image and determine the best regularization parameters for image fidelity and resolution.

The MPoL software used in this study is open source and designed with ALMA measurement sets in mind. Anyone wishing to use MPoL for RML imaging of ALMA data should follow these general steps.

1. **Obtain arrays of complex visibility data.** MPoL is designed to work directly with arrays of complex visibilities in order to avoid potential issues with installation and version control

of CASA and related tools. Visibilities can be obtained from CASA measurement sets using CASA tools. There are also open source software packages like visread[8] that can aid in this process.

2. **Select pixel size and number of pixels in the image.** These pixels will serve as the model parameterization, and allow loose visibilities to be gridded. It is important to be mindful of the choice of pixel size, as pixels that are too large will place an intrinsic resolution limit on the final image, and using too many small pixels will introduce an unnecessary computational burden on the imaging process. At this point, it is also a good idea to examine the dirty image in MPoL to make sure that everything has been loaded and initialized as expected.

3. **Set the initial state of the model and determine which regularizers to include in the loss function.** We recommend initializing the model with the dirty image. Dirty image initialization leads to faster convergence of the loss function. The loss function may include any number of regularizing terms.

4. **Define a range of hyperparameter values to be tested with CV.** We recommend beginning with a coarse grid of values. The hyperparameter values that yield the highest fidelity images can change with the dataset, so starting with a wide range of values (i.e. spanning several orders of magnitude) helps quickly hone in on a narrower range of potential values. For example, while one dataset might minimize the CV scores with an entropy prior with $\lambda = 0.1$, another dataset might minimize the CV score with $\lambda = 10$, so a coarse round of CV might test $\lambda = [0.01, 0.1, 1, 10]$.

5. **Set up CV process by defining the number of K-folds and data partitioning scheme.** We recommend using $K = 10$, which has been shown to balance bias and variance in the estimation and has already been used with success in interferometric imaging.

6. **Perform CV (perhaps in a coarsely-defined round and a fine-tuning round) to obtain optimal imaging hyperparameters.** The set of hyperparameters that minimizes the CV score correspond to the model with the best predictive performance. Ensure that each K-fold has reached convergence during the CV process; if each K-fold did not fully converge, the CV score is invalid and CV must be restarted with enough iterations to allow full convergence.

7. **Image using the full dataset with the hyperparameters that minimized the CV score.** Ensure that the model has fully converged. If it has, the result is a fully regularized image.

While this is a rough outline of a functional RML workflow, in practice the RML imaging process need not be so linear. For some imaging cases, one may find it beneficial to try individual regularizers before combining them, or to produce some preliminary images before running the full CV process. When selecting regularizers, we recommend considering the qualitative features the source is likely to have. Maximum entropy and sparsity regularization tend to produce high-resolution features, and sparsity has the added benefit of effectively removing background noise. TV and TSV regularizers

[8] https://mpol-dev.github.io/visread/

tend to emphasize sharp edges in the image, with TV enforcing sharp edges more rigidly. These general characteristics can help inform which regularizers to include in the loss function.

Overall, RML techniques provide flexible imaging processes that are well-suited for applications to ALMA measurement sets, particularly observations of protoplanetary disks. Future work will explore regularization on a wider range of source morphologies observed by ALMA, including image cubes with many velocity channels. We will also probe additional types of regularization, such as regularization across velocity channels for spectral line data, or regularization on quantities derived from data such as power spectrum regularization.

REFERENCES

Akiyama, K., Ikeda, S., Pleau, M., et al. 2017a, AJ, 153, 159, doi: 10.3847/1538-3881/aa6302

Akiyama, K., Kuramochi, K., Ikeda, S., et al. 2017b, ApJ, 838, 1, doi: 10.3847/1538-4357/aa6305

ALMA Partnership, Brogan, C. L., Pérez, L. M., et al. 2015, ApJL, 808, L3, doi: 10.1088/2041-8205/808/1/L3

Andrews, S. M., Wilner, D. J., Zhu, Z., et al. 2016, ApJL, 820, L40, doi: 10.3847/2041-8205/820/2/L40

Andrews, S. M., Huang, J., Pérez, L. M., et al. 2018, ApJL, 869, L41, doi: 10.3847/2041-8213/aaf741

Bishop, C. M. 2006, Pattern Recognition and Machine Learning, ed. M. Jordan, J. Kleinberg, & Schölkopf, Bernhard (Springer-Verlag New York). https://www.springer.com/gb/book/9780387310732

Breiman, L., & Spector, P. 1992, International Statistical Review / Revue Internationale de Statistique, 60, 291. http://www.jstor.org/stable/1403680

Cárcamo, M., Román, P. E., Casassus, S., Moral, V., & Rannou, F. R. 2018, Astronomy and Computing, 22, 16, doi: 10.1016/j.ascom.2017.11.003

Carrillo, R. E., McEwen, J. D., & Wiaux, Y. 2012, MNRAS, 426, 1223, doi: 10.1111/j.1365-2966.2012.21605.x

—. 2014, MNRAS, 439, 3591, doi: 10.1093/mnras/stu202

Chael, A. A., Johnson, M. D., Bouman, K. L., et al. 2018, ApJ, 857, 23, doi: 10.3847/1538-4357/aab6a8

Chael, A. A., Johnson, M. D., Narayan, R., et al. 2016, ApJ, 829, 11, doi: 10.3847/0004-637X/829/1/11

Cornwell, T. J., & Evans, K. F. 1985, A&A, 143, 77

Deisenroth, M. P., Faisal, A. A., & Ong, C. S. 2020, Mathematics for Machine Learning (Cambridge University Press). https://www.cambridge.org/us/academic/subjects/computer-science/pattern-recognition-and-machine-learning/mathematics-machine-learning

Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2019, ApJL, 875, L4, doi: 10.3847/2041-8213/ab0e85

Hastie, T., Tibshirani, R., & Friedman, J. 2009, The Elements of Statistical Learning (Springer-Verlag New York), doi: 10.1007/978-0-387-84858-7

Högbom, J. A. 1974, A&AS, 15, 417

Honma, M., Akiyama, K., Uemura, M., & Ikeda, S. 2014, PASJ, 66, 95, doi: 10.1093/pasj/psu070

Huang, J., Andrews, S. M., Dullemond, C. P., et al. 2018, ApJL, 869, L42, doi: 10.3847/2041-8213/aaf740

Kohavi, R. 1995, in

Kuramochi, K., Akiyama, K., Ikeda, S., et al. 2018, ApJ, 858, 56, doi: 10.3847/1538-4357/aab6b5

Li, F., Cornwell, T. J., & de Hoog, F. 2011, A&A, 528, A31, doi: 10.1051/0004-6361/201015045

McMullin, J. P., Waters, B., Schiebel, D., Young, W., & Golap, K. 2007, in Astronomical Society of the Pacific Conference Series, Vol. 376, Astronomical Data Analysis Software and Systems XVI, ed. R. A. Shaw, F. Hill, & D. J. Bell, 127

Molinaro, A. M., Simon, R., & Pfeiffer, R. M. 2005, Bioinformatics, 21, 3301, doi: 10.1093/bioinformatics/bti499

Murphy, K. P. 2012, Machine Learning: A Probabilistic Perspective (The MIT Press). https://mitpress.mit.edu/books/machine-learning-1

Narayan, R., & Nityananda, R. 1986, ARA&A, 24, 127, doi: 10.1146/annurev.aa.24.090186.001015

Paszke, A., Gross, S., Massa, F., et al. 2019, in Advances in Neural Information Processing Systems 32, ed. H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Curran Associates, Inc.), 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pérez, S., Casassus, S., Baruteau, C., et al. 2019, AJ, 158, 15, doi: 10.3847/1538-3881/ab1f88

Rudin, L. I., Osher, S., & Fatemi, E. 1992, Physica D Nonlinear Phenomena, 60, 259, doi: 10.1016/0167-2789(92)90242-F

Sivia, D. S., & Skilling, J. 2006, Data Analysis - A Bayesian Tutorial, 2nd edn., Oxford Science Publications (Oxford University Press)

Tibshirani, R. 1996, Journal of the Royal Statistical Society. Series B (Methodological), 58, 267. http://www.jstor.org/stable/2346178

Wiaux, Y., Puy, G., & Vandergheynst, P. 2010, MNRAS, 402, 2626, doi: 10.1111/j.1365-2966.2009.16079.x

Yamaguchi, M., Akiyama, K., Tsukagoshi, T., et al. 2020, ApJ, 895, 84, doi: 10.3847/1538-4357/ab899f