Draft version March 13, 2019
Typeset using LaTeX **modern** style in AASTeX62

Image Forward Modeling Techniques

1. NOTATION

I try to use script letters for probability distributions (e.g. $\mathcal{S}$), with parameters as greek letters (e.g. γ). Draws from a distribution are denoted by $\sim$. Functions are given uppercase italic letters (e.g. Q). Scalar variables are lowercase italic letters. Vector variables are ~~bolded~~ lowercase. Matrices are ~~bolded~~ uppercase.

2. INTRODUCTION

We want a generative model for the image(s) that can be plugged into a likelihood function and an MC sampling scheme. Therefore the goal is to go from the scene to counts in pixels. The basic function that we are trying to compute (efficiently) is:

$$\hat{C}_n = F_n(\gamma, \beta) \tag{1}$$

where $\hat{C}_n$ is the expected counts in the nth pixel, there are $n = 1, 2, ...N$ pixels, γ are the parameters describing the instrument (PSF, PRF, distortions, etc.), and β are the parameters describing the scene (i.e. source positions and shapes). In the end we will want to calculate

$$\mathcal{L}(\{C_n\} \,|\, \{t_{\exp} \hat{C}_n\}) = \prod_n \frac{e^{-t_{\exp} \hat{C}_n} \, (t_{\exp} \hat{C}_n)^{C_n}}{C_n!} \tag{2}$$

(or probably the Gaussian likelihood since there will be plenty of counts.) Ideally we will also be able to efficiently calculate (analytically) the gradients of the ln-likelihood with respect to any of the source (and perhaps even instrument) parameters. Analytic gradients can vastly accelerate both optimization and MCMC bayesian inference.

3. FULL PROBLEM

What's really happening from a photon's point of view (sort of)? First it comes from a distribution $\mathcal{S}(\alpha, \delta[, \nu] \,|\, \beta)$ which we take to be the intensity distribution on the sky, or the *scene*, parameterized by β. It then travels through the the atmosphere (if there is one), then the telescope optics, finally reaching the detector plane. During this travel the photon is diffracted, so that its position is now in iteself a PDF (called the point spread function, $\mathcal{P}$). Upon striking the detector the wavefunction is collapsed and the detector position is drawn from this PDF, with some probability related to the quantum efficiency of the detector at that position (and the reflectivity of the optics, etc.). One must also consider gaps in the detector, and intrapixel sensitivity variations, which might be folded into a spatially dependent QE or pixel response function, or alternatively combined with the PSF somehow. The mapping from the celestial position to the detector plane depends on the distortion of the optics and the aforemention probabilistic draw from $\mathcal{P}$. This can all be represented by

$$\begin{aligned}
\alpha_j, \delta_j[, \nu_j] &\sim \mathcal{S}(\beta) \\
x_j, y_j &= D(\alpha_j, \delta_j) \\
\Delta x_j, \Delta y_j &\sim \mathcal{P}(\gamma, x_j, y_j[, \nu_j]) \\
\hat{C}_n &\propto \lim_{J\to\infty} \frac{1}{J} \sum_{j=1}^{J} Q_n(x_j + \Delta x_j, y_j + \Delta y_j[, \nu_j])
\end{aligned} \tag{3}$$

where $\mathcal{S}$ is the sky instensity distribution in celestial coordinates and depends on parameters β, α_j and δ_j are the coordinates of the jth draw from $\mathcal{S}$, D is the mapping from celestial to detector coordinates (x, y) including distortion, $\mathcal{P}$ is the point-spread-function in detector coordinates and depends on parameters γ as well as the detector location, $\Delta x_j, \Delta y_j$ are draws from $\mathcal{P}$, $Q_n(x, y)$ is the detector quantum efficiency of the nth pixel at location (x, y) (usually something close to a bivariate boxcar), and $\hat{C}_n$ is the expected countrate in pixel n per unit time and total number of photons due to celestial sources (i.e. excluding detector background.) The scene can be expressed as a sum of individual source distributions $\mathcal{S} = \sum_{m=1}^{M} \mathcal{S}_m$. In what follows we will drop ν from consideration, assuming that $Q(\nu)$ is a constant that can be factored out.

In the limit of infinite exposure time (or very many photons), we can replace the sum of a large dumber of draws with integrals to obtain the expected count rate.

$$\begin{aligned}
\mathcal{M}(x, y \,|\, \gamma, \beta) &= \iint dx'\, dy'\, \mathcal{P}(x - x', y - y' \,|\, \gamma, x', y')\, \mathcal{S}(D^{-1}(x', y') \,|\, \beta) \\
\hat{C}_n &= \iint_{A_n} dx\, dy\, Q_n(x, y)\, \mathcal{M}(x, y \,|\, \gamma, \beta)
\end{aligned} \tag{4}$$

where A_n is the region of the nth pixel, $\mathcal{M}$ is the PSF-convolved scene, $\mathcal{P}$ is the point-spread function (which depends on the input position of the photon), and D is the mapping from sky coordinates to detector coordinates, including distortions. One question is whether it makes more sense to think of the pixels as rectangles in detector coordinates and project the scene into these coordinates as we have done above, or to project the pixel boundaries (and $\mathcal{P}$ and Q) into celestial coordinates. This probably depends on the form of D and on how the Q and $\mathcal{P}$ functions are determined. Nevertheless, by switching the order of integration we can write

$$\begin{aligned} \mathcal{R}_n(x', y' \,|\, \gamma) &= \iint_{A_n} dx\, dy\, Q_n(x, y)\, \mathcal{P}(x' - x, y' - y \,|\, \gamma, x, y) \\ \hat{C}_n &= \iint dx'\, dy'\, \mathcal{R}_n(x', y' \,|\, \gamma)\, \mathcal{S}(D^{-1}(x', y') \,|\, \beta) \end{aligned} \tag{5}$$

where $\mathcal{R}_n(x', y')$ gives the response of a pixel to a point source at detector coordinates (x', y') including the effect of the pixel-averaged PSF. This form can be easier to compute, since this pixel response function (PRF) can be specified beforehand, or at least once approximated includes the annoying integral over A_n.

These convolutions and integrals can be analytically intractable, or time consuming. This is especially true the case of complicated forms for $\mathcal{P}$, $\mathcal{R}$, $\mathcal{S}$ or the function D. Much of the guts of any forward modeling algorithm will be concerned with approximating this integral, especially in a way that yields analytic gradients.

4. APPROXIMATION BY GAUSSIAN MIXTURES (AND SUBGRIDDING)

One method to make the integrals more tractable is to calculate $\mathcal{R}$ and $\mathcal{S}$ on fine grids and calculate 5 directly on this grid. The grid has to be fine enough that the integral can be approximated by a sum; that is, the grid must fully sample the PRF and the scene.

$$\hat{C}_{n'} \approx \sum_{i,j} \mathcal{R}_{n'}(x_i, y_j \,|\, \gamma)\, \mathcal{S}(D^{-1}(x_i, y_j) \,|\, \beta)$$

It can then be rebinned to produce the actual pixel $\hat{C}_n$. One approach to obtain gradients with respect to the parameters γ, β is to further approximate $\mathcal{R}$ and $\mathcal{S}$ by mixtures of Gaussians. This allows one to write the countrate as

$$\begin{aligned} \mathcal{R}_{n'}(x, y \,|\, \gamma) &= \sum_k w_k(\gamma)\, \mathcal{N}(x, y \,|\, \mu_{k,n'}(\gamma), \Sigma_k(\gamma)) \\ \mathcal{S}(x, y \,|\, \beta) &= \sum_\ell w_\ell(\beta)\, \mathcal{N}(D^{-1}(x, y) \,|\, \mu_\ell(\beta), \Sigma_\ell(\beta)) \\ \hat{C}_{n'} &= \sum_{i,j} \sum_k w_k\, \mathcal{N}(x_i, y_j \,|\, \mu_{k,n'}, \Sigma_{k,n'})\, \sum_\ell w_\ell\, \mathcal{N}(D^{-1}(x_i, y_j) \,|\, \mu_\ell, \Sigma_\ell) \end{aligned} \tag{6}$$

where $\mathcal{N}(a, b \,|\, \mu, \Sigma)$ is the bivariate normal distribution with location μ and covariance matrix Σ evaluated at (a, b).

The drawbacks here are: 1) evaluating large numbers of exponentials 2) finding nice representations of parameterized scenes and PRFs in terms of mixtures of Gaussians such that the gradients of the Gaussian parameters with respect to the scene parameters β are relatively simple 3) limited accuracy of the mixture of Gaussians approximation. 4) Gaussians do not have a defined maximum frequency and are thus always undersampled to some degree by any grid. 5) Location dependent PSF or PRF can be unwieldy. One could also replace the $\mathcal{R}$ mixture of Gaussians with the polynomial or spline approximation discussed below. I think the Tractor does something like 4 but with a δ-function Q, in which case the PSF and the PRF are equivalent.

4.1. *Undersampling*

One difficulty in this method is accounting for undersampling of the PRF by the pixels. The JWST F090W has pixels $\sim$ FWHM of the PSF. One strategy is to make calculations on subpixel grids and then sum/rebin. However, this entails quite a bit of extra computation (and gradients have to be tracked). Another option is to account for the undersampling by using a second order approximation to the integral of the gaussian over a square pixel. In Figure 1 there is a demonstration of the accuracy of this apprximation compared to a highly oversampled and rebinned calculation, for a particular subpixel shift and a gaussian PSF with FWHM of 1 pixel. If this proves insufficiently accurate, it would not be that difficult (except in compute time) to implement a mild oversampling scheme to improve the accuracy.

Here is the math for the 2nd-order approximation to a gaussian centered at (x_s, y_s) with amplitude A_s and covariance matrix Σ_s

$$\begin{aligned} I(x_p, y_p) &= A_s\, h\, e^g && (7)\\ \Delta_x, \Delta_y &= x_p - x_s, y_p - y_s \\ g &= -\frac{1}{2}\left(\Delta_x V_x + \Delta_y V_y\right) \\ h &= 1 + \frac{1}{24}\left(V_x^2 + V_y^2 - F_{xx} - F_{yy}\right) \\ V_x &= F_{xx}\,\Delta_x + F_{xy}\,\Delta_y \\ V_y &= F_{yy}\,\Delta_y + F_{xy}\,\Delta_x \\ F &= \Sigma^{-1} \end{aligned} \quad (8)$$

where

$$F = \begin{pmatrix} F_{xx} & F_{xy} \\ F_{xy} & F_{yy} \end{pmatrix}$$

4.2. *Method for Approximating PSF with Gaussians*

We've been using an EM algorithm that is fast for gaussian mixtures, implemented by Jerry Anunrojwong (Harvard). The gaussian mixtures are fit to the oversampled PSF images (normalized to unity). The likelihood function is a standard χ^2 where the variance of each PSF pixel is proportional to the pixel value. The number of gaussians to use is specified before fitting (we can scan over number of gaussians to choose the best number). Centers, sizes, and ellipticities of each gaussian are allowed to vary (i.e. are fitted). The result of running this on the F090W oversampled PSF image with 6 Gaussians is shown in figure 2. The resulting parameters of the gaussian are then converted to units of true (rather than oversampled) image pixels.

4.3. *Method for Approximating Sersic profiles with Gaussians*

Following Hogg & Lang (2013), but we: 1) use absolute celestial units (e.g. mas); 2) fix the radii of the gaussians (on sky), including one at $r = 0$; 3) apply a small blurring to the Sersic profile and Gaussians before fitting (using analytic Sersic blurring formulae from Trujillo et al. (2001)); and 4) Fit for the amplitudes as a function of both n_{sersic} and half-light radius r_h. 5) Regularize the amplitudes to be smooth as a function of Gaussian radii for a given n_{sersic} and r_h

We make these changes for two reasons. First because we don't want to be calculating tons of gaussians below a scale that matters for the PRF. Second because the fixed radii of the Gaussians and the smoothness of the amplitude variations will allow us to spline the amplitudes as a function of n_{sersic} and r_h for easy derivatives, so that we can fit for n.

This all results in a fixed number M of Gaussians for each profile, with amplitudes $a_m(n, r_h)$, zero mean, and covariances $\Sigma_{m,i,j} = \delta_{i,j}\, r_m^2$ where here $\delta_{i,j}$ is the Kroneker-δ and r_m is the dispersion of the mth Gaussian (in mas or something like it). We also re-normalize so that

$$\sum_{m}^{M} a_m = 1 \tag{9}$$

An example of the Sersic, smoothed Sersic, and Gaussian approximation thereof is given in Figure 3.

4.4. *Astrometric Distortions*

Multiple images (with different astrometry) will be used to constrain a single source. Therefore need to be able to convert between celestial coordinates and on-sky parameters (e.g. the half-light radius in arcseconds, the position angle on the sky) to image or pixel coordinates. We will make the approximation that for any region of interest the astrometric distortion can be approximated by an affine transformation. Indeed this will be part of the definition of of coherent region of interest.

The standard FITS style astrometry for such a situation (Greisen et al. 2002) is something like

$$z = T\,(\mu_p - \mu_0) \tag{10}$$

where z are intermediate world coordinates, μ_p are the pixel coordinates $(x,\,y)$, μ_0 gives an image or region dependent reference pixel and T is the region dependent transformation matrix (often represented in FITS headers by `CD_ij` entries). T need not have a determinant of 1, but it must be non-singular.

We will define

$$\begin{aligned} D &= T^{-1} \qquad (11)\\ &= \begin{pmatrix} \frac{\partial p_1}{\partial z_1} & \frac{\partial p_1}{\partial z_2} \\ \frac{\partial p_2}{\partial z_1} & \frac{\partial p_2}{\partial z_2} \end{pmatrix} \qquad (12)\\ \mu_p &= D\,z + \mu_0 \\ d &= \sqrt{||D||} \end{aligned}$$

and use the matrix D going forward. The scalar d is a dilation factor, which would be important if we defined our fixed radii (or rather fixed dispersion) galaxy Gaussians in the pixel space. Note that for a given subregion of a particular exposure the matrix D and the reference coordinates μ_0, c_0 are fixed.

Now, we also need the transformation form celestial coordinates $c = (\alpha, delta)$ to intermediate world coordinates. In general this is non-linear, as the transformation itself depends on the value of c. We will write

$$W = \begin{pmatrix} \frac{\partial z_1}{\partial \alpha} & \frac{\partial z_1}{\partial \delta} \\ \frac{\partial z_2}{\partial \alpha} & \frac{\partial z_2}{\partial \delta} \end{pmatrix} \tag{13}$$

$$\tag{14}$$

and W_0 the value of this matrix at some reference position c_0, typically the center of a postage stamp. A good approximation is

$$W = \begin{pmatrix} \cos(\delta_0) & 0 \\ 0 & 1 \end{pmatrix} \tag{15}$$

$$\tag{16}$$

Then the transformation that we need is something like

$$\mu_p = \mu_0 + D\, W_0\, (c - c_0) \tag{17}$$

4.5. *Shapes*

There's a question about how best to parameterize galaxy shapes. We have already addressed the overall scale by including r_h as a parameter controlling the amplitudes in our Gaussian mixture Sersic approximation. Now we consider the two other common parameters, axis ratio $q = \frac{b}{a}$ (or inclination angle θ) and position angle φ. These can be described by rotation and scaling matrices

$$R = \begin{pmatrix} \cos(\varphi) - \sin(\varphi) \\ \sin(\varphi) \;\; \cos(\varphi) \end{pmatrix} \tag{18}$$

$$S = \begin{pmatrix} \frac{1}{\sqrt{q}} & 0 \\ 0 & \sqrt{q} \end{pmatrix} \tag{19}$$

where we have used $\sqrt{q}$ so that$||S|| = 1$. (In fact, in the forcepho code we have redifined $q \equiv (b/a)^{1/2}$ when using this parameterization). These transformations have the problem that the position angle becomes unconstrained as q goes to 1. In terms of a posterior distribution, this leads to a funnel shape in the q, φ space, which can be difficult to sample from.

The lensing community has explored alternative parameterizations to describe shear that do not suffer from this problem. In particular Bernstein, & Jarvis (2002) describes the following parameterization in terms of the *conformal shear* vector $\boldsymbol{\eta}$

$$q = \frac{b}{a} \tag{20}$$

$$\boldsymbol{s} = (b/a, \varphi) \tag{21}$$

$$q = e^{-\eta} \tag{22}$$

$$\eta_+ = \eta\, \cos(2\varphi) \tag{23}$$

$$\eta_\times = \eta\, \sin(2\varphi) \tag{24}$$

$$\boldsymbol{\eta} = (\eta_+, \eta_\times) \tag{25}$$

$$\tag{26}$$

The vector $\boldsymbol{\eta}$ is a vector in a non-Euclidean space. The vector $(\eta_+, 0)$ creates ellipses oriented to the x or y axes while the vector $(0, \eta_\times)$ orients ellipses along axes rotated 45^{deg} from the coordinate axes. For reasons that will become clear later, it will be useful to find the Jacobian of the transformation from $\boldsymbol{s} = q, \varphi$ to $\boldsymbol{\eta} = \eta_+, \eta_\times$.

	q	φ
η_+	$-\frac{\cos(2\varphi)}{q}$	$2\,\ln q\,\sin(2\varphi)$
$\eta_\times$	$-\frac{\sin(2\varphi)}{q}$	$-2\,\ln q\,\cos(2\varphi)$

And in fact we will usually want the inverse of this matrix (which has determinant $\frac{2\ln q}{q}$)

$$J = \begin{pmatrix} -q\,\cos(2\varphi) & -q\,\sin(2\varphi) \\ \frac{\sin(2\varphi)}{2\ln q} & -\frac{\cos(2\varphi)}{2\ln q} \end{pmatrix} \tag{27}$$

$$d\boldsymbol{s} = J\,d\boldsymbol{\eta} \tag{28}$$

This might need to be transposed. Also an additional factor of $\frac{1}{2\sqrt{q}}$ is necessary in the code to account for the non-standard definition of q in the code.

4.6. *Analytic convolutions and transformations to pixel space*

Assuming that we have represented the galaxy and the PRF as mixtures of Gaussians we can write the convolution of the galaxy with the PRF as

$$I(\mu_p) = \sum_{m=1}^{M}\sum_{\ell=1}^{L} \psi\; a_m\, a_\ell\, \mathcal{N}(\mu_p \,|\, \mu_m + \mu_\ell, \Sigma'_m + \Sigma_\ell) \tag{29}$$

where ψ is the total source flux, μ_p is the pixel central coordinates, the a are the normalized amplitudes of the Gaussian, Σ'_m and μ_m describes a galaxy Gaussian rotated and stretched into the image plane, and Σ_ℓ and μ_ℓ describe a PSF gaussian. For a single term, we can write more explicitly

$$I_{m,\ell}(\mu_p) = A\;\exp(-0.5\,(\mu_p - \mu_{m,\ell})^{\mathsf{T}}\,F\,(\mu_p - \mu_{m,\ell})) \tag{30}$$

$$\begin{aligned} A &= \frac{\psi\,a_m\,a_\ell\,\sqrt{||F||}}{2\pi} \\ \Sigma &= (\Sigma'_m + \Sigma_\ell) \\ F &= \Sigma^{-1} \end{aligned} \tag{31}$$

$$\begin{aligned} \mu_{m,\ell} &= \mu_m + \mu_\ell \\ \mu_m &= D\,(c - c_0) + \mu_0 \\ \Sigma'_m &= D\,R\,S\,\Sigma_m\,S^{\mathsf{T}}\,R^{\mathsf{T}}\,D^{\mathsf{T}} \end{aligned}$$

where S is the scale matrix parameterized by the (on-sky) axis ratio q, R is the rotation matrix parameterized by the (on-sky) position angle of the major axis φ, D is the distortion matrix to go from on-sky to pixel coordinates, and Σ_m is the covariance matrix describing the mth Gaussian in the untransformed (circularly symmetric) on-sky coordinate space. Note that $||R|| = ||S|| = 1$. Also $||D||$ is constant.

The Gaussian in pixel space has 6 parameters ϕ_g: the amplitude $A = (\psi\, a_m\, a_\ell \sqrt{||F||})/(2\pi)$, the two parameters of the mean $\mu_{m,\ell} = x_{m,\ell}, y_{m,\ell}$, and the three parameters of the inverse covariance matrix F_{xx}, F_{yy}, and $F_{xy} = F_{yx}$.

There are 7 on-sky galaxy source parameters θ_s: the total flux ψ, the celestial coordinates $c = (\alpha, \delta)$, the Sersic index n, the half-light radius in celestial units (e.g. degrees or milliarcseconds) r_h, the axis ratio $q = (b/a)$, and position angle φ. In general we will want to construct the Jacobian between ϕ_g and θ_s for each Gaussian.

Now, one option to deal with the galaxy size parameter is to let the size of each galaxy Gaussian be set by the scale matrix S, so that the whole collection of Gaussians grows by replacing the matrix S above with $S' = s\, S$ where s is a scalar size. In this scheme the amplitudes a_m would be functions only of n. However, the radii of the Gaussians would vary. However, if for any reason we have made the absolute scale important in our approximation of the Sersic profile by Gaussians (for example by adding a sub-PRF blurring before fitting the profiles, as described above) then this will be problematic and internally inconsistent. We may also want to use fixed radii for other reasons. In this case the size of the of the galaxy can come in through the amplitudes only, and the a_m will be functions of both n and r_h.

4.7. *Gradients*

Gradients of the likelihood with respect to the source parameters are useful for optimization and certain efficient kinds of MCMC sampling. The gradient in the likelihood is simply related to the gradient in in the pixel counts. We can calculate gradients of the pixel counts with respect to the image plane Gaussian parameters $\partial I/\partial\phi$ fairly easily.

$$\frac{\partial I}{\partial A_s} = h\, e^g \tag{32}$$

$$\frac{\partial I}{\partial x_s} = I\, V_x - \frac{A_s\, e^g}{12}\,(F_{xx}\, V_x + F_{xy}\, V_y) \tag{33}$$

$$\frac{\partial I}{\partial y_s} = I\, V_y - \frac{A_s\, e^g}{12}\,(F_{yy}\, V_y + F_{xy}\, V_y) \tag{34}$$

$$\frac{\partial I}{\partial F_{xx}} = -\frac{I\,\Delta_x^2}{2} - \frac{A_s\, e^g}{24}\,(1 + 2\,\Delta_x\, V_x) \tag{35}$$

$$\frac{\partial I}{\partial F_{yy}} = -\frac{I\,\Delta_y^2}{2} - \frac{A_s\, e^g}{24}\,(1 + 2\,\Delta_y\, V_y) \tag{36}$$

$$\frac{\partial I}{\partial F_{xy}} = -I\,\Delta_x\,\Delta_y - \frac{A_s\, e^g}{24}\,(\Delta_y\, V_x + \Delta_x\, V_y) \tag{37}$$

But we will also need the Jacobian to go from the image plane Gaussian parameters ϕ to the galaxy parameters θ. The table below gives $\partial\phi_i/\partial\theta_j$ which can be matrix multiplied with the $\partial I/\partial\phi_i$ vector:

	ψ	α	δ	q	φ	n	r_h
A	$\frac{K}{\psi}$	–	–	∇_7	∇_8	$\frac{K}{a_m}\frac{\partial a_m}{\partial n}$	$\frac{K}{a_m}\frac{\partial a_m}{\partial r_h}$
x	–	D_{00}	D_{01}	–	–	–	–
y	–	D_{10}	D_{11}	–	–	–	–
F_{xx}	–	–	–	∇_1	∇_4	–	–
F_{xy}	–	–	–	∇_2	∇_5	–	–
F_{yy}	–	–	–	∇_3	∇_6	–	–

where

$$\begin{aligned}
K &= \frac{\psi\, a_\ell\, a_m\, \sqrt{||F||}}{2\pi} \\
\left(\begin{smallmatrix}\nabla_1 & \nabla_2 \\ \nabla_2 & \nabla_3\end{smallmatrix}\right) &= \frac{\partial F}{\partial q} \\
\left(\begin{smallmatrix}\nabla_4 & \nabla_5 \\ \nabla_5 & \nabla_6\end{smallmatrix}\right) &= \frac{\partial F}{\partial \varphi} \\
\nabla_7 &= \frac{K}{2\,||F||}\frac{\partial ||F||}{\partial q} \\
\nabla_8 &= \frac{K}{2\,||F||}\frac{\partial ||F||}{\partial \varphi} \\
\frac{\partial ||F||}{\partial \theta} &= ||F||\ \mathrm{Tr}(\Sigma\,\frac{\partial F}{\partial \theta}) \\
\frac{\partial F}{\partial \theta} &= -F\,\frac{\partial \Sigma}{\partial \theta}\,F \\
\frac{\partial \Sigma}{\partial q} &= D\,R\,\frac{\partial S}{\partial q}\,\Sigma_m\,S^\mathsf{T}\,R^\mathsf{T}\,D^\mathsf{T} + D\,R\,S\,\Sigma_m\,\frac{\partial S^\mathsf{T}}{\partial q}\,R^\mathsf{T}\,D^\mathsf{T} \\
\frac{\partial \Sigma}{\partial \varphi} &= D\,\frac{\partial R}{\partial \varphi}\,S\,\Sigma_m\,S^\mathsf{T}\,R^\mathsf{T}\,D^\mathsf{T} + D\,R\,S\,\Sigma_m\,S^\mathsf{T}\,\frac{\partial R^\mathsf{T}}{\partial \varphi}\,D^\mathsf{T}
\end{aligned} \tag{38}$$

5. PARAMETER TRANSFROMATIONS

Sometimes it is useful to sample in parameters other than the native galaxy parameters θ. This is true when using MCMC samplers that expect the parameters to be unconstrained. In this case, it is possible to sample in an unconstrained parameter

z and then use a logistic transform to convert to constrained parameters. However, because we still desire the prior to be defined (uniform by default) in the constrained parameter space, we must account for this change of variables when calculating the posterior probability. And, of course, the ln-posterior gradients with respect to the parameters θ must be multiplied by the Jacobian of this transformation, J_b

$$\theta_i = g(z_i) \tag{39}$$
$$= \theta_{min} + (\theta_{max} - \theta_{min}) \frac{1}{1+e^{-z_i}} \tag{40}$$
$$J_{b,i} = \frac{\partial \theta_i}{\partial z_i} \tag{41}$$
$$= R\, g(z_i)\,(1 - g(z_i)) \tag{42}$$
$$R \equiv \theta_{max} - \theta_{min} \tag{43}$$

The term that needs to be added to the ln-posterior probability is the natural log of the absolute value of the determinant of the Jacobian (basically this is a parameter volume correction.) For the simple independent parameter transfromations described above, this is easily calculated as

$$\ln P = \ln \mathcal{L} + \ln Pr + \ln |||J_b||| \tag{44}$$
$$= \ln \mathcal{L} + \ln Pr + \ln |\prod_i \frac{\partial \theta_i}{\partial z_i}| \tag{45}$$
$$\tag{46}$$

Now the gradients of the ln-posterior become somewhat more complicated, since this Jacobian term also ahs to be differentiated. Fortunately the logistic function we are using has easy second derivatives.

$$\frac{\ln |\partial||J_b|||}{\partial z_i} = \frac{1}{|||J_b|||} \frac{||J_b||}{|||J_b|||} \frac{\partial ||J_b||}{\partial z_i} \tag{47}$$
$$= \frac{1}{|||J_b|||} \frac{||J_b||}{|||J_b|||} \frac{||J_b||}{J_{b,i}} \frac{\partial^2 \theta_i}{\partial^2 z_i} \tag{48}$$
$$= \frac{1}{|||J_b|||} \frac{||J_b||}{|||J_b|||} \frac{||J_b||}{J_{b,i}} J_{b,i}\,(1 - 2\,g(z_i)) \tag{49}$$
$$= (1 - 2\,g(z_i)) \tag{50}$$

assuming the determinant is always real.

6. TRANSLATION INTO CODE

There are two main kinds of classes. One is PostageStamps, which contain the image data and information about the image. This includes the distortion matrix and the

PSF (as a set of gaussians). PostageStamps are assumed to have a single constant PSF and a linear coordinate transformation. The other is Sources, which can be PointSources or Sersic profiles, and contain information about the source parameters. There is then code which translates the Sources and PostageStamp properties into on-image collections of Gaussians and the associated Jacobians, and another set of code that computes and accumulates the pixel counts and gradients thereof.

6.1. *Accumulation*

We need to accumulate the on-image gradients correctly into the vector of all parameters. We try to use the notation above, but make some small changes.

First, we write the vector of all on-sky scene parameters (for all sources) as

$$\Theta \in \mathbb{R}^S \tag{51}$$
$$= (\theta_j)_{j=1}^{j=J} \tag{52}$$
$$\theta_j \in \mathbb{R}^{6+B} \tag{53}$$
$$S = (6+B)\,J \tag{54}$$

where each source has on-sky parameters θ_j, and there are B different bands.

Then, for a given subimage or stamp k we can write the sequence of parameters of on-image gaussians corresponding to a single source in that image as

$$G = (\phi_{j,k,\ell,m}) \tag{55}$$
$$\phi_{j,k,\ell,m} \in \mathbb{R}^6 \tag{56}$$

6.2. *Levels of parallelization*

There are various places we could parallelize this code, some of them more amenable to GPU or MIC parallelization:

- at the sample level (k)
- at the pixel level (n)
- at the source level (m) (this is probably a bad idea, since number of sources may not be well matched to the number of processors, and may be variable.)
- at the image level

7. APPROXIMATION BY POLYNOMIALS AND SAMPLING

One way to approximate intractable integrals is through Monte Carlo techniques. This has the nice property that the precision of the approximation can be dialed up or down by increasing or decreasing the number of samples. In this scheme we represent each component of the scene as a set of draws from $\mathcal{S}_m(\alpha, \delta \mid \beta_m)$, which we call phonions. In order to maintain differentiability with respect to the parameters β, the draws can be fixed in some latent space $\hat{\alpha}, \hat{\delta}$, possibly with some associated weight $\hat{w}$, and then transformed to the actual space via affine transformations

$$\begin{aligned}\hat{\alpha}, \hat{\delta}[, \hat{w}] &\sim \mathcal{S}(\beta = \beta_0) \\ \alpha_k, \delta_k &= F(\hat{\alpha}, \hat{\delta}, \beta) \\ w_k &= G(\hat{w}, \beta) \\ x_k, y_k &\sim D(\alpha_k, \delta_k)\end{aligned} \tag{57}$$

where F is an affine transformation. This has easy derivatives, iff D, and G have easy derivatives.

One could then also draw samples from $\mathcal{P}$ and apply these to the scene samples, following 3, but this might result in unwanted sampling noise unless the number of samples is very large (which is inefficient). Furthermore the derivatives of $Q(x, y)$ with respect to x and y are likely to have discontinuities. Alternatively, we can use a polynomial approximation to $\mathcal{R}$ to approximate the integral as

$$\begin{aligned}\mathcal{R}_n(x_k, y_k) &\approx \sum_{i,j} R_{n,i,j}\,(x_k - x_n)^i\,(y_k - y_n)^j \\ \hat{C}_n &\approx \sum_k w_k\,\mathcal{R}_n(x_k, y_k \mid \gamma)\end{aligned} \tag{58}$$

where x_n, y_n are the coordinates of the pixel center or some other suitable reference coordinate for that pixel, and the $R_{n,i,j}$ are coefficients to a polynomial approximation of $\mathcal{R}$. This has the benefit of making $\mathcal{R}$ analytically differentiable with respect to x, y (which in turn depend on the parameters β. We could also use a mixture of Gaussians approximation for $\mathcal{R}$ described above, but the polynomial (or a spline) approximation has some nice properties.

7.1. *Gradients*

How do we do gradients in this scheme? We want

$$\begin{aligned}\frac{\partial \hat{C}_n}{\partial \beta} &\approx \sum_k \frac{\partial \mathcal{R}_n(x_k, y_k \mid \gamma)}{\partial \beta} \\ &\approx \sum_k \sum_{i,j=0} R_{n,i,j} \frac{\partial w_k}{\partial \beta}\, i\, j\, \frac{\partial x_k}{\partial \beta} \frac{\partial y_k}{\partial \beta} (x_k - x_n)^{i-1}\,(y_k - y_n)^{j-1}\end{aligned} \tag{59}$$

7.2. *Draws from the Scene*

There are a frew ways to consider drawing from the scene. Perhaps the closest to the photon case would be to draw directly from some bivariate flux distribution. This can be accomplished by calculating the inverse of the bivariate cumulative flux distribution (CFD), and then transforming uniform (or regular) sampling in the cumulative flux distribution to x, y or r, ϕ pairs. However, because the flux distributions are usually circularly symmetric in the latent space, we can separate into a CFD as a function of r with a uniform distribution in ϕ.

Here's what that would look like for a Sersic profile with scale length of 1 (we can change this and the ellipticity through affine transformations) and Sersic index η.

$$\begin{aligned} p(r) &= \frac{2\pi}{f_{\text{total}}} r e^{-r^{1/\eta}} \\ p(z) &= \frac{2\pi\eta}{f_{\text{total}}} z^{2\eta-1} e^{-z} \\ f_{\text{total}} &= 2\pi\eta\Gamma(2\eta) \\ CFD(r) &= \int_0^r p(r')\,dr' \\ &= \gamma(2\eta, r^{1/\eta})/\Gamma(2\eta) \qquad (60) \\ & \qquad (61) \end{aligned}$$

where $\Gamma(n)$ is the gamma function and $\gamma(n, z)$ is the incomplete gamma function. Transforming uniform or regular draws from the CFD into values for R thus requires the inverse incomplete gamma function. However, this transformation depends on the value of η. Thus everytime a new η value is considered, we will need to redo the transformation. In this case the latent variables are actually the draws from the CFD. So if we want to vary η in the fit then the gradients of this transformation with respect to η. Unfortunately there is no way to move between different Sersic indices via only affine transformations of any latent space. This means we lose the nice quality of keeping all of our transformations simple (i.e. without expensive transcendental functions).

The requirement that we re-transform the CFD draws in draws in R above leads to some additional complexity for derivative calculation, since we now need to add another level to the chain rule. Alternatively, we can associate *weights* to each draw from a fiducial latent CFD. In this case, instead of re-transforming the CFD draws into R draws, we can simply reweight the draws based on the ratio of the desired p(R) to the fiducial p(R) at that R.

8. OTHER CODES

Other codes that do all or some of generative modeling of exposure level images plus inference of joint or marginalized posteriors for source parameters include

- Photo (SDSS/HSC/LSST). Optimization of parametric models.
- The Tractor (DECaLs, MzLS). Optimization (and posterior sampling) of multi-band parameteric models.
- Celeste (SDSS)
- PROFIT
- GALFIT/GALAPAGOS
- IMFIT
- BUDDA
- SCARLET
- PCAT
- DAOPHOT
- Dolphot
- MOPEX
- Mofongo? (CANDELS, 3DHST)
- TPHOT (CANDELS, DEEP)
- XID+ (Herschel)
- GalSim (Euclid) photon shooting
- UFIG photon shooting
- PhoSim (LSST) photon shooting
- Guitarra (NIRCAM) photon shooting

REFERENCES

Bernstein, G. M., & Jarvis, M. 2002, AJ, 123, 583.

Greisen et al. 2002, A&A 395, 1061

Hogg, D. W. & Lang, D. 2013, PASP 127, 719

Trujillo, I. et al. 2001, MNRAS 321, 269

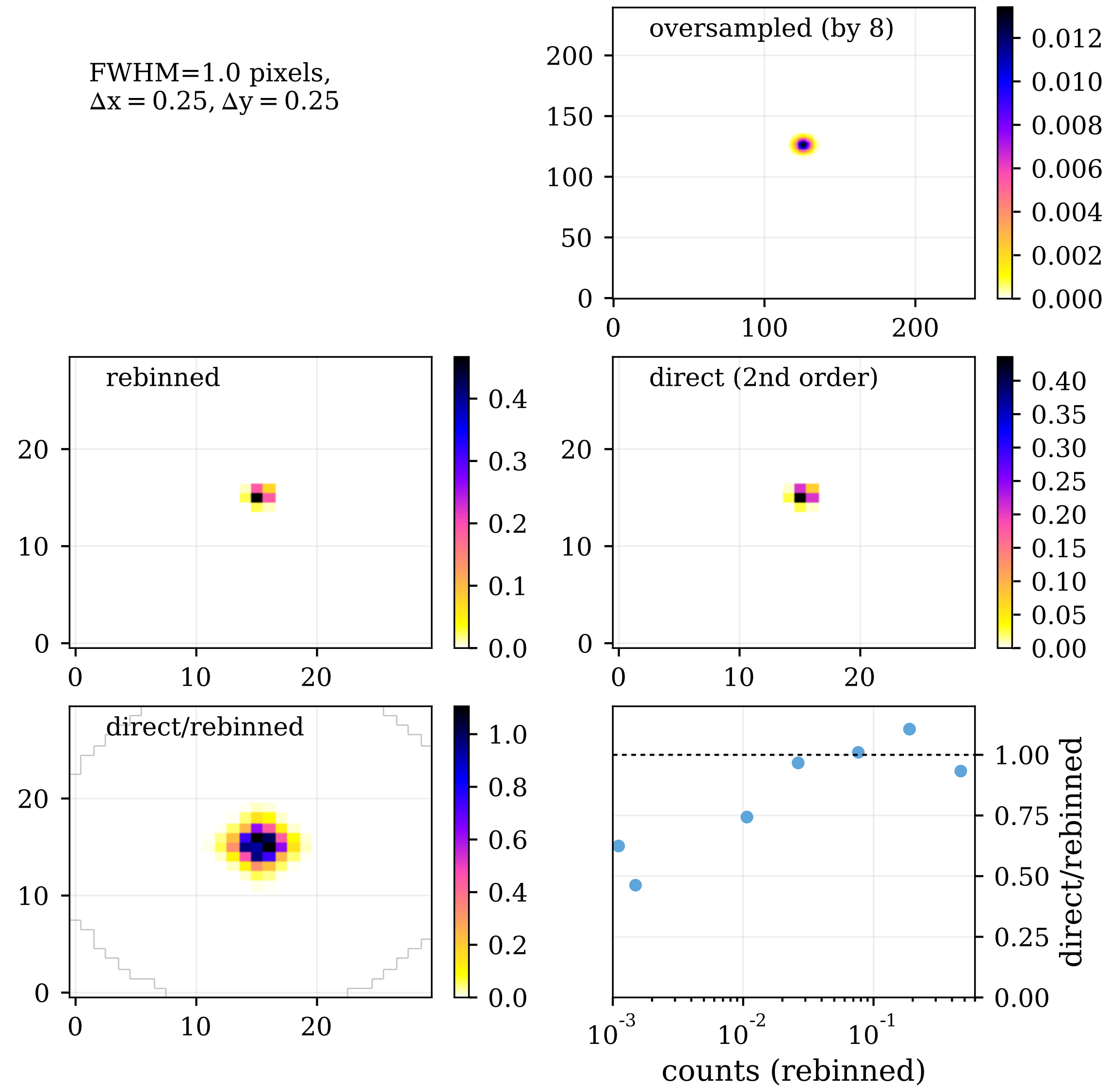

Figure 1. A demonstration of the effect of undersampling using a symmetric gaussian PSF with FWHM=1.0 native detector pixels (similar to the undersampling of the F090W PSF). The PSF is shifted 0.25 pixels in both x and y. *Top Right:* The PSF calculated on a grid oversampled from the native detector pixels by a factor of 8. *Middle left:* The oversampled PSF rebinned to the native detector pixel grid, giving an accurate reperesentation of the true pixel counts that would be expected for this PSF. *Middle right:* The PSF calculated directly on the native grid using the 2nd-order approximation to the pixel value. *Bottom Left:* The ratio of the direct 2nd order calculation to the oversampled and rebinned pixels fluxes, as a function of pixel flux.

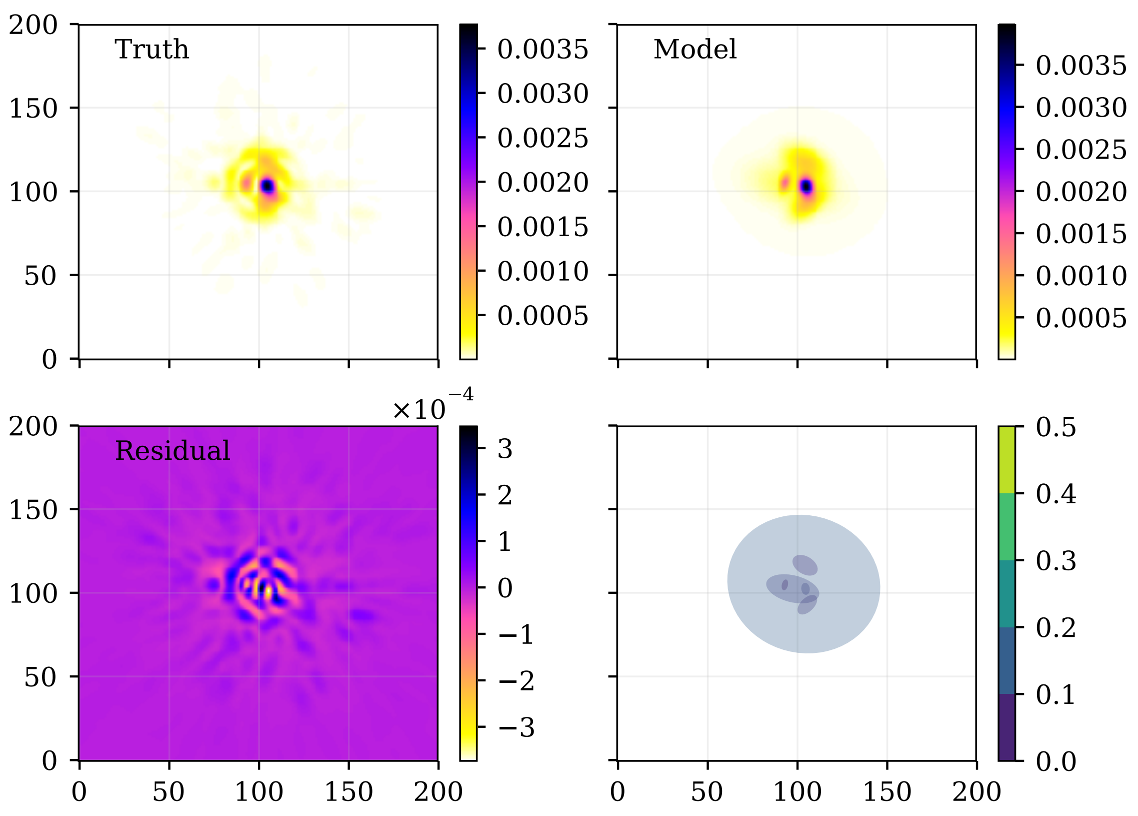

Figure 2. A mixture of 6 Gaussians fit to theF090W PSF. *Top Left:* The input image of the PSF (oversampled by a factor of 8 from the detector pixels) *Top Right:* The fitted model PSF. *Bottom Left:* The residual, showing the model PSF subtracted from the input PSF *Bottom Right:* Ellipses showing the location, sizes, and position angle of the 6 fitted Gaussians, color coded by the total fractional flux of each Gaussian.

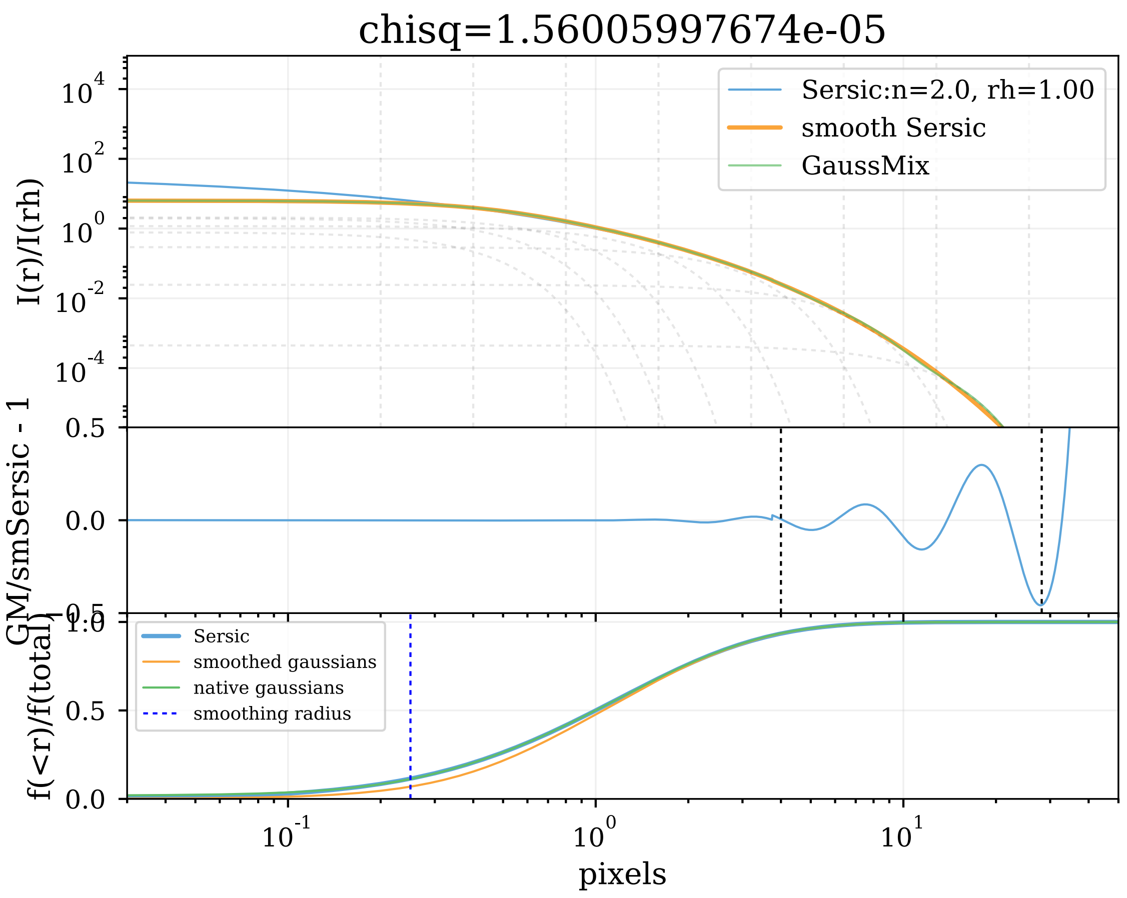

Figure 3. A mixture of Gaussians fit to a smoothed Sersic model. *Top:* Radial surface brightness profiles of the unsmoothed Sersic profile for $n = 2, r_h = 1$ pixel (*blue*), the Sersic profile smoothed with Gaussian of 0.25 pixels dispersion (*orange*), and the Gaussian mixture approximation (*green*). The vertical dashed lines show the fixed dispersion of each of the zero mean Gaussians used in the approximation, and the dashed curves show the contribution of each Gaussian to the approximated profile. *Middle:* The fractional profile residual. *Bottom:* The cumulative flux as a function of radiaus for each profile.

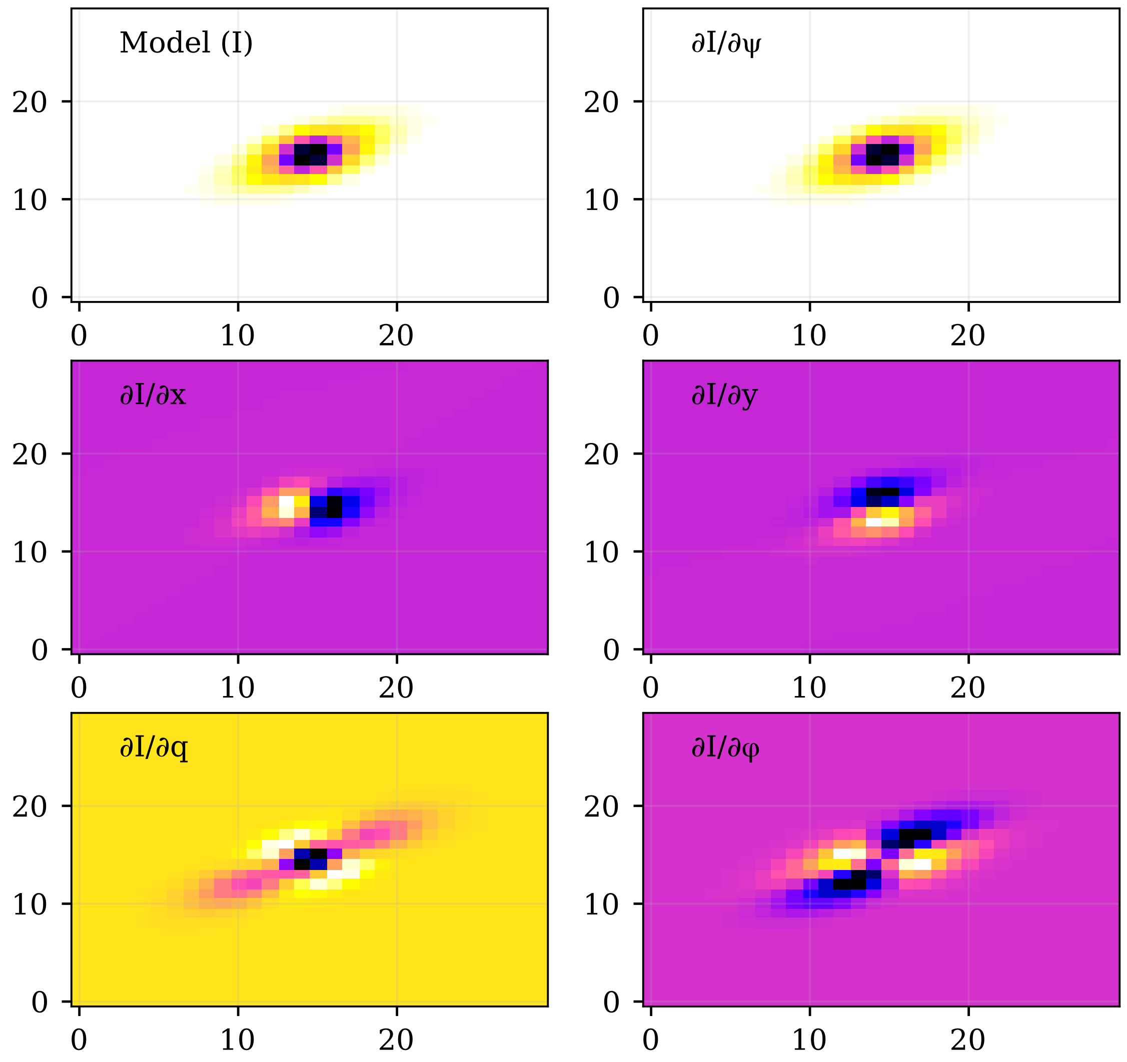

Figure 4. For a single gaussian PSF and a simple single Gaussian model (with axis ratio $q = 0.5$ and position angle $\varphi = 30^{\rm deg}$), this figure shows the model and the image gradients with respect to flux, x, y, q, and φ.

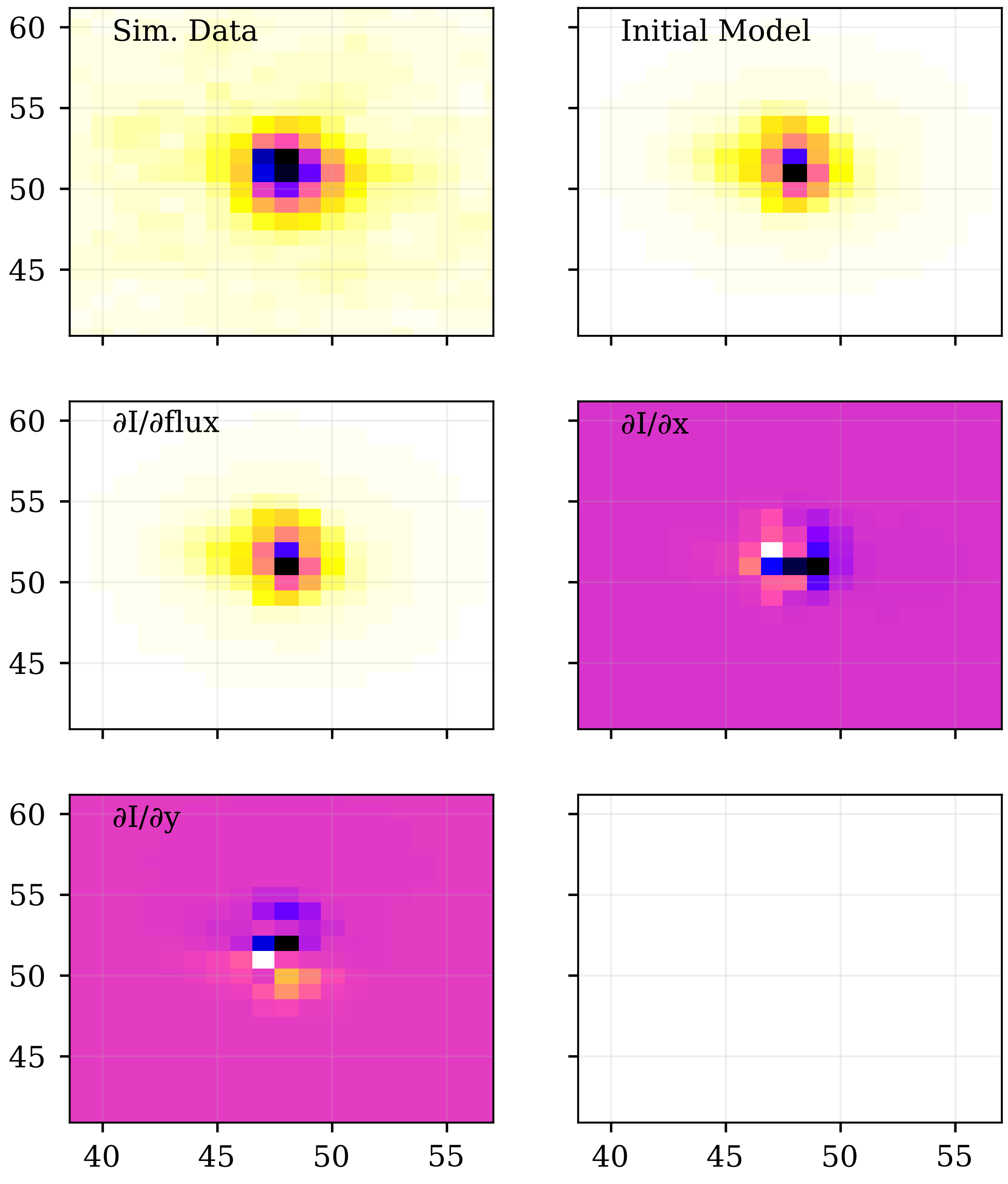

Figure 5. *Top left:* Simulated F090W point source data from CW. *Top Right:* Initial model guess. *Succeeding Panels:* Gradient images of the initial model with respect to the 3 point source model parameters.