

IMFIT: A FAST, FLEXIBLE NEW PROGRAM FOR ASTRONOMICAL IMAGE FITTING

PETER ERWIN^{1,2}

Draft version August 7, 2014

ABSTRACT

I describe a new, open-source astronomical image-fitting program called IMFIT, specialized for galaxies but potentially useful for other sources, which is fast, flexible, and highly extensible. A key characteristic of the program is an object-oriented design which allows new types of image components (2D surface-brightness functions) to be easily written and added to the program. Image functions provided with IMFIT include the usual suspects for galaxy decompositions (Sérsic, exponential, Gaussian), along with Core-Sérsic and broken-exponential profiles, elliptical rings, and three components which perform line-of-sight integration through 3D luminosity-density models of disks and rings seen at arbitrary inclinations.

Available minimization algorithms include Levenberg-Marquardt, Nelder-Mead simplex, and Differential Evolution, allowing trade-offs between speed and decreased sensitivity to local minima in the fit landscape. Minimization can be done using the standard χ^2 statistic (using either data or model values to estimate per-pixel Gaussian errors, or else user-supplied error images) or the Cash statistic; the latter is particularly appropriate for cases of Poisson data in the low-count regime. I show that fitting low-S/N galaxy images using χ^2 minimization can lead to significant biases in fitted parameter values, which are avoided if the Cash statistic is used; this is true even when Gaussian read noise is present.

Subject headings: methods: data analysis — techniques: image processing — techniques: photometric — galaxies: structure — galaxies: bulges — galaxies: photometry

1. INTRODUCTION

Galaxies are morphologically complex entities. Even seemingly simple systems like elliptical galaxies can have outer envelopes and distinct cores or nuclei, while so-called “bulgeless” spiral galaxies can still have nuclear star clusters and disks with complex radial or vertical profiles. In order to accurately describe the structure of galaxies, it is often necessary to decompose galaxies into component substructures. Even single-component systems are often modeled with analytic functions in order to derive quantitative measurements such as scale lengths or half-light radii, Sérsic indices, etc.

The traditional method for dealing with this complexity has been to model 1D surface-brightness profiles of galaxies – derived from 2D images – as the sum of separate, additive components (e.g., bulge + disk). While this 1D approach can be conceptually and computationally simple, it has a number of limitations, above and beyond the fact that it involves discarding most of the data contained in an image. To begin with, there are uncertainties about *what type* of 1D profile to use – should one use major-axis cuts or profiles from ellipse fits to isophotes, should the independent variable be semi-major axis or mean radius, etc. It is also difficult to correctly account for the effects of image resolution when fitting 1D profiles; attempts to do so generally require simple analytic models of the point-spread function (PSF), extensive numerical integrations, and the assumption of circular symmetry for the PSF, the surface-brightness function, or both (e.g., Pritchet & Kline 1981; Saglia et al. 1993; Trujillo et al. 2001; Rusli et al. 2013). Furthermore, there are often intrinsic degeneracies involved: images of galaxies with non-axisymmetric components such as bars can yield 1D profiles resembling those from galaxies

with axisymmetric bulges, which makes for considerable ambiguity in interpretation. Finally, if one is interested in the properties of non-axisymmetric components (bars, elliptical rings, spiral arms) themselves, it is generally impossible to extract these from 1D profiles.

A better approach in many cases is to directly fit the images with 2D surface-brightness models. Early approaches along this line include those of Capaccioli et al. (1987), Shaw & Gilmore (1989), and Scorza & Bender (1990). The first general, self-consistent 2D bulge+disk modeling of galaxy images – that is, constructing a full 2D model image, comparing its intensity values with the observed image pixel-by-pixel, and iteratively updating the parameters until the χ^2 is minimized – was that of Byun & Freeman (1995), with de Jong (1996) being the first to include extra, non-axisymmetric components (bars) in fitting galaxy images. An interesting alternate approach developed at roughly the same time was the Multi-Gaussian Expansion method (Monnet et al. 1992; Emsellem et al. 1994; Cappellari 2002), which involves modeling both PSF and image as the sum of an arbitrary number of elliptical Gaussians; the drawback is the difficulty that lies in trying to associate sets of Gaussians with particular structural components and parameters.

The most commonly used galaxy-fitting codes at the present time are probably GIM2D (Simard 1998; Simard et al. 2002),³ GALFIT (Peng et al. 2002, 2010),⁴ BUDDA (de Souza et al. 2004; Gadotti 2008),⁵ and MGE (Emsellem et al. 1994; Cappellari 2002).⁶ GIM2D is specialized for bulge-disk decompositions and is implemented as an IRAF package, using the Metropolis algorithm to minimize the total χ^2 for models containing an exponential disk and a Sérsic bulge. BUDDA is

¹ Max-Planck-Institut für extraterrestrische Physik, Giessenbachstrasse, 85748 Garching, Germany

² Universitäts-Sternwarte München, Scheinerstrasse 1, 81679 München, Germany

³ <https://www.astrosoci.ca/users/GIM2D/>

⁴ <http://users.obs.carnegiescience.edu/peng/work/galfit/galfit.html>

⁵ <http://www.sc.eso.org/~dgadotti/budda.html>

⁶ <http://www-astro.physics.ox.ac.uk/mxc/software/>

written in FORTRAN and is also specialized for bulge-disk decompositions, though it includes a wider variety of possible components: exponential disk (with optional double-exponential profile), Sérsic bulge, Sérsic bar, analytic edge-on disk, and nuclear point source. It uses a version of the Nelder-Mead simplex method (Nelder & Mead 1965), also known as the “downhill simplex”, for χ^2 minimization. GALFIT, which is written in C, is the most general of these codes, since it allows for arbitrary combinations of components (including components with different centers, which allows the simultaneous fitting of overlapping galaxies) and includes the largest set of possible components; the latest version (Peng et al. 2010) includes options for spiral and other parametric modulation of the basic components. GALFIT uses a version of the fast Levenberg-Marquardt gradient-search method (Levenberg 1944; Marquardt 1963) for its χ^2 minimization. MGE, available in IDL and Python versions, is rather different from the other codes in that it uses what is effectively a non-parametric approach, fitting images using the sum of an arbitrary number of elliptical Gaussians (it is similar to GALFIT in using the Levenberg-Marquardt method for χ^2 minimization during the fitting process.)

For most astronomical image-fitting programs the source code is not generally available, or else is encumbered by non-open-source licenses. Even when the code *is* available, it is not easy to extend the built-in sets of predefined image components. The simplest codes provide only elliptical components with exponential and Sérsic surface brightness profiles; more sophisticated codes such as BUDDA and (especially) GALFIT provide a larger set of components, including some sophisticated ways of perturbing the components in the case of GALFIT. But if one wants to add completely new functions, this is not easy. (The case of MGE is somewhat different, since it does not allow parametric functions at all.)

As an example of why one might want to do this, consider the case of edge-on (or nearly edge-on) disk galaxies. Both BUDDA and GALFIT include versions of the analytical solution for a perfectly edge-on, axisymmetric, radial-exponential disk of van der Kruit & Searle (1981), with a sech² function for the vertical light distribution. But real galaxy disks are not always perfectly edge-on, do not all have single-exponential radial structures, and their vertical structure may in some cases be better described by a sech or exponential profile, or something in between (e.g., van der Kruit 1988; de Grijs et al. 1997; Pohlen et al. 2004; Yoachim & Dalcanton 2006). Various authors studying edge-on disks have suggested that models using radial profiles other than a pure exponential would be best fit via line-of-sight integration through 3D luminosity-density models (e.g., van der Kruit & Searle 1981; Pohlen et al. 2000, 2004). More sophisticated approaches could even involve line-of-sight integrations that account for scattering and absorption by dust (e.g., Xilouris et al. 1997, 1998, 1999).

Another potential disadvantage of existing codes is that they rely on the Gaussian approximation of Poisson statistics for the fitting process. While this is eminently sensible for dealing with many CCD and near-IR images, it can produce biases when applied to images with low count rates (see Humphrey et al. 2009 and Section 9 of this paper). This is why packages for fitting X-ray data, such as Sherpa (Freeman et al. 2001), often include alternate statistics for fits.

In this paper, I present IMFIT, a new, open-source image-fitting code designed to overcome some of the limitations mentioned above. In particular, IMFIT uses an object-oriented

design which makes it relatively easy to add new, user-designed image components; it also provides multiple fitting algorithms and statistical approaches. It can also be extremely fast, since it is able to take advantage of multiple CPU cores on the same machine to execute calculations in parallel.

The outline of this paper is as follows. Section 2 provides a quick sketch of how the program works, while Section 3 details the process of generating model images and the configuration files which describe the models. The different underlying statistical models and minimization algorithms used in the fitting process are covered in Section 4; methods for estimating confidence intervals for fitted parameters are discussed in Section 5. The default 2D image functions which can be used in models are presented in Section 6; this includes functions which perform line-of-sight integration through 3D luminosity-density models (Section 6.2). After a brief discussion of coding details (Section 7), two examples of using IMFIT to model galaxy images are presented in Section 8: the first involves fitting a moderately-inclined spiral galaxy with disk, bar, and ring components, while the second fits an edge-on spiral galaxy with thin and thick edge-on disk components. Finally, Section 9 discusses possible biases to fitted parameters when the standard χ^2 statistic is used in the presence of low-count images, using both model images and real images of elliptical galaxies.

2. GENERAL OUTLINE OF THE PROGRAM

IMFIT begins by processing command-line options and then reads in the data image, along with any optional, user-specified PSF, noise, and mask images (all in FITS format). The configuration file is also read; this specifies the model which will be fit to the data image, including initial parameter values and parameter limits, if any (see Section 3.1).

The program then creates an instance of the `ModelObject` class, which holds the relevant data structures, instances of the image functions specified by the configuration file, and the general code necessary for computing a model image. If χ^2 minimization (the default) is being done, a noise image is constructed, either from a user-specified FITS file already read in or by internally generating one, assuming the Gaussian approximation for Poisson noise. The noise image is then converted to $1/\sigma^2$ form and combined with the mask image, if any, to form a final weight image used for calculating the χ^2 value. (If model-based χ^2 minimization has been specified, then the noise image, which is based on the model image, is recalculated and combined with the mask image every time a new model image is computed; if the Cash statistic is being used for minimization, then no noise image is read or created and the weight image is constructed directly from the mask image. See Section 4.1 for more on the different statistical approaches.)

The actual fitting process is overseen by one of three possible nonlinear minimization algorithms, as specified by the user. These algorithms proceed by generating or modifying a set of parameter values and feeding these values to the aforementioned model object, which in turn calculates the corresponding model image, convolves it with the PSF (if PSF convolution is part of the model), and then calculates the fit statistic (e.g., χ^2) by comparing the model image with the stored data image. The resulting fit statistic is returned to the minimization algorithm, which then updates the parameter values and repeats the process according to the details of the particular method, until the necessary stop criterion is reached –

e.g., no further significant reduction in the fit statistic, or a maximum number of iterations. Finally, a summary of the fit results is printed to the screen and saved to a file, along with any additional user-requested outputs (final model image, final residual image, etc.).

3. CONSTRUCTING THE MODEL IMAGE

3.1. Configuration File

The model which will be fit to the data image is specified by a configuration file, which is a text file with a relatively simple and easy-to-read format; see Figure 1 for an example.

The basic format for this file is a set of one or more “function blocks”, each of which contains a shared center (pixel coordinates) and one or more image functions. A function block can, for example, represent a single galaxy or other astronomical object, which itself has several individual components (e.g., bulge, disk, bar, ring, nucleus, etc.) specified by the individual image functions. Thus, for a basic bulge/disk decomposition the user could create a function block consisting of a single Sérsic function and a single Exponential function. There is, however, no a priori association of any particular image function or functions with any particular galaxy component, nor is there any requirement that a single object must consist of only one function block. The final model is the sum of the contributions from all the individual functions in the configuration file. The number of image functions per function block is unlimited, and the number of function blocks per model is also unlimited.

Each image function is listed by name (e.g., “FUNCTION Sérsic”), followed by the list of its parameters. For each parameter, the user supplies an initial guess for the value, and (optionally) either a comma-separated, two-element list of lower and upper bounds for that parameter or the keyword “fixed” (indicating that the parameter will remain constant during the fit).⁷

The total set of all individual image-function parameters, along with the central coordinates for each function block, constitutes the parameter vector for the minimization process.

3.2. Image Functions

An image function can be thought of as a black box which accepts a set of parameter values for its general setup, and then accepts individual pixel coordinates (x, y) and returns a corresponding computed intensity (i.e., surface brightness) value for that pixel. The total intensity for a given pixel in the model image (prior to any PSF convolution) is the sum of the individual values from each image function.

This design means that the main program needs to know nothing about the individual image functions except the number of parameters they take, and which subset of the total parameter vector corresponds to a given image function. The actual calculations carried out by an image function can be as simple or as complex as the user requires, ranging from returning a constant value for each pixel (e.g., the FlatSky function) to performing line-of-sight integration through a 3D luminosity density model (e.g., the ExponentialDisk3D function); user-written image functions could even perform modest simulations in the setup stage.⁸

⁷ For minimizations using the Differential Evolution algorithm, the initial values are actually ignored, but the parameter bounds are *required*; see Section 4.3.

⁸ One should bear in mind that even relatively simple fits will typically

The list of currently available image functions, along with descriptions for each, is given in Section 6.

3.3. PSF Convolution

To simulate the effects of atmospheric seeing and telescope optics, model images can be convolved with a PSF image. The latter can be any FITS file which contains the point spread function. PSF images should be square, with the peak of the PSF centered in the image. (Off-center PSFs can be used, but the resulting convolved model images will of course be shifted.) IMFIT automatically normalizes the PSF image when it is read in.

The actual convolution follows the standard approach of using Fast Fourier Transforms of the internally-generated model image and the PSF image, multiplied together, with the output convolved model image being the inverse transform of the product image. The transforms are done with the FFTW library (“Fastest Fourier Transform in the West”, Frigo & Johnson 2005),⁹ which has the advantage of being able to perform transforms on images of arbitrary size (i.e., not just images with power-of-two sizes); in addition, it is well-tested and fast, and can use multiple threads to take advantage of multiple processor cores.

3.4. Makeimage: Generating Model Images Without Fitting

A companion program called MAKEIMAGE is included in the IMFIT package, built from the same codebase as IMFIT itself. This program implements the complete model-image construction process, including PSF convolution, and then simply saves the resulting model image as a FITS file. It can optionally save separate images, one for each of the individual image functions that make up the model. Since it uses the same configuration-file format as IMFIT, it can use the output best-fit parameter file that IMFIT produces (or even an input IMFIT configuration file).

It also has an optional mode which estimates the fractional flux contributions of the individual components in the model, by summing up the total flux of the individual components on a pixel-by-pixel basis using a very large internal image (by default, 5000×5000 pixels). Although analytic expressions for total flux exist for some common components, this is not true for all components – and one of the goals of IMFIT is to allow users to create and use new image functions without worrying about whether they have simple analytic expressions for the total flux. This mode can be used to help determine such things as B/T and other ratios after a fit is found, although it is up to the user to decide which of the components is the “bulge”, which is the “disk”, and so forth.

4. THE FITTING PROCESS

4.1. The Statistical Background and Options

Given a vector of parameter values $\vec{\theta}$, a model image is generated with per-pixel predicted data values m_i , which are then compared with the *observed* per-pixel data values d_i . The goal is to find the $\vec{\theta}$ which produces the best match between m_i and d_i , subject to the constraints of the underlying statistical model.

The usual approach is based on the maximum-likelihood principle (which can be derived from a Bayesian perspective

require dozens to hundreds of function evaluations during the minimization process, so complex simulations will mean a lengthy fitting process.

⁹ <http://www.fftw.org>

```

# Comment lines:  all lines beginning with "#" are ignored.
# Anything following a "#" on a line is ignored, too.

# Optional keywords describing the image
GAIN 2.7      # A/D gain for image in e/ADU
READNOISE 4.5  # image read-noise in electrons

# This is the first function block, which has 2 image functions
# (1 Sersic + 1 exponential)
# It begins with the starting guess for pixel coordinates of center (x,y)
# Each parameter has initial value, then (optional) lower,upper bounds

X0  150.1    148,152
Y0  149.5    148,152
FUNCTION Sersic      # A 2D elliptical Sersic function
PA  95.0     0,180   # major-axis position angle [deg]
ell 0.05     0,1     # ellipticity
n   2.5      0.5,4.0  # Sersic index
I_e 20.0     # intensity [counts/pixel] at r_e
r_e 5.0      # half-light radius in pixels
FUNCTION Exponential # A 2D elliptical exponential function
PA  95.0     0,180
ell 0.45     0,1
I_0 90.0     fixed   # this parameter will be held fixed
h   15.0     # (no bounds for this parameter)

# This is the second function block:  just a single exponential

X0  225.0    224,226
Y0  181.7    180,183
FUNCTION Exponential
PA  22.0     0,180
ell 0.25     0,1
I_0 10.0
h   20.0

```

Figure 1. Example of a configuration file for IMFIT. Comments are colored red.

if, e.g., one assumes constant priors for the parameter values). To start, one considers the per-pixel likelihood $p_i(d_i|m_i)$, which is the probability of observing d_i given the model prediction m_i and the underlying statistical model for how the data are generated.

The goal then becomes finding the set of model parameters which maximizes the total likelihood \mathcal{L} , which is simply the product over all N pixels of the individual per-pixel likelihoods:

$$\mathcal{L} = \prod_{i=1}^N p_i. \quad (1)$$

It is often easier to work with the logarithm of the total likelihood, since this converts a product over pixels into a sum over pixels, and can also simplify the individual per-pixel terms. As most nonlinear optimization algorithms are designed to *minimize* their objective function, one can use the *negative* of the log-likelihood. Thus, the goal of the fitting process be-

comes minimization of the following:

$$-\ln \mathcal{L} = -\sum_{i=1}^N \ln p_i. \quad (2)$$

During the actual minimization process, this can often be further simplified by dropping any additive terms in $\ln p_i$ which do not depend on the model, since these are unaffected by changes in the model parameters and are thus irrelevant to the minimization.

In many circumstances, multiplying the negative log-likelihood by 2 produces a value which has the property of being distributed like the χ^2 distribution (e.g., Cash 1979, and references therein); thus, it is conventional to minimize $-2\ln \mathcal{L}$.

4.1.1. The (Impractical) General Case: Poisson + Gaussian Statistics

The data in astronomical images typically consist of detections of individual photons from the sky + telescope system

(including photons from the source, the sky background, and possibly thermal backgrounds in the telescope) in individual pixels, combined with possible sources of noise due to read-out electronics, digitization, etc.

Photon-counting statistics obey the Poisson distribution, where the probability of detecting x photons per integration, given a true rate of m , is

$$P(x) = \frac{m^x e^{-m}}{x!}. \quad (3)$$

Additional sources of (additive) noise such as read noise tend to follow Gaussian statistics with a mean of 0 and a dispersion of σ , so that the probability of measuring d counts after the readout process, given an input of x counts from the Poisson process, is

$$P(d) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(d-x)^2}{2\sigma^2}\right]. \quad (4)$$

The general case for most astronomical images thus involves both Poisson statistics (for photon counts) and Gaussian statistics (for read noise and other sources of additive noise). Unfortunately, even though the individual elements are quite simple, the combination of a Gaussian process acting on the output of a Poisson process leads to the following rather frightening per-pixel likelihood (e.g., Llacer & Nuñez 1991; Nuñez & Llacer 1993):

$$p_i(d_i|m_i) = \sum_{x_i=0}^{\infty} \frac{m_i^{x_i} e^{-m_i}}{x_i!} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(d_i-x_i)^2}{2\sigma^2}\right). \quad (5)$$

The resulting negative log-likelihood for the total image (dropping terms which do not depend on the model) is

$$-\ln \mathcal{L} = \sum_{i=1}^N \left(m_i - \ln \left[\sum_{x_i=0}^{\infty} \frac{m_i^{x_i}}{x_i!} \exp\left(-\frac{(d_i-x_i)^2}{2\sigma^2}\right) \right] \right). \quad (6)$$

Since this still contains an infinite series of exponential and factorial terms, it is clearly rather impractical for fitting images rapidly.

4.1.2. The Simple Default: Pure Gaussian Statistics

Fortunately, there is a way out which is appropriate for many astronomical images. This is to use the fact that the Poisson distribution approaches a Gaussian distribution when the counts become large. In this approximation the Poisson distribution is replaced by a Gaussian with $\sigma = \sqrt{m}$. It is customary to assume this is valid when the counts are $\gtrsim 20$ per pixel (e.g., Cash 1979), though Humphrey et al. (2009) point out that biases in the fitted parameters can be present even when counts are higher than this; see Section 9 for examples in the case of 2D fits.

Since the contribution from read noise is also nominally Gaussian, the two can be added in quadrature, so that the per-pixel likelihood function is just

$$p_i(d_i|m_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left[-\frac{(d_i-m_i)^2}{2\sigma_i^2}\right], \quad (7)$$

where $\sigma_i^2 = \sigma_{m_i}^2 + \sigma_{\text{RN}}^2 = m_i + \sigma_{\text{RN}}^2$, with σ_{RN} being the dispersion of the read-noise term. Twice the negative log-likelihood of the total problem then becomes (dropping terms which do

not depend on the model) the familiar χ^2 sum:

$$-2\ln \mathcal{L} = \chi^2 = \sum_{i=1}^N \frac{(d_i-m_i)^2}{\sigma_i^2}. \quad (8)$$

This is the default approach used by IMFIT: minimizing the χ^2 as defined in Eqn. 8.

The approximation of the Poisson contribution to σ_i is based on the model intensity m_i . Traditionally, it is quite common to estimate this from the *data* instead, so that $\sigma_i^2 = \sigma_{d_i}^2 + \sigma_{\text{RN}}^2 = d_i + \sigma_{\text{RN}}^2$. This has the nominal advantage of only needing to be calculated once, at the start of the minimization process, rather than having to be recalculated every time the model is updated.¹⁰ However, the bias resulting from using data-based errors in the low-count regime can be worse than the bias introduced by using model-based σ_i values (see Section 9). Both approaches are available in IMFIT, with data-based σ_i estimation being the default. The data-based and model-based approaches are often referred to as “Neyman’s χ^2 ” and “Pearson’s χ^2 ”, respectively; in this paper I use the symbols χ_d^2 and χ_m^2 to distinguish between them.

In the case of “error” images generated by a data-processing pipeline, the corresponding σ_i or σ_i^2 (variance) values can easily be used in Equation 8 directly, under the assumption that the final per-pixel error distributions are still Gaussian.

4.1.3. The Simple Alternative: Pure Poisson Statistics

So why not always use the Gaussian χ^2 approximation, as is done in most image-fitting packages?

In the absence of any noise terms except Poisson statistics – something often true of high-energy detectors, such as X-ray imagers – the individual-pixel likelihoods are just the probabilities of a Poisson process with mean m_i , where the probability of recording d_i counts is

$$p_i(d_i|m_i) = \frac{m_i^{d_i} e^{-m_i}}{d_i!} \quad (9)$$

This leads to a very simple version of the negative log-likelihood, often referred to as the “Cash statistic” C , after its derivation in Cash (1979):

$$-2\ln \mathcal{L} = C = 2 \sum_{i=1}^N (m_i - d_i \ln m_i), \quad (10)$$

where the factorial term has been dropped because it does not depend on the model.

Humphrey et al. (2009) point out that using the Cash statistic is preferable even when the counts are above the nominal limit of ~ 20 per pixel, since fitting pure-Poisson data using the χ^2 Gaussian approximation can lead to biases in the derived model parameters. Section 9 presents some examples of this effect using both artificial and real galaxy images, and shows that the effect persists even when moderate (Gaussian) read noise is *also* present.

Using the Cash statistic is also appropriate when fitting simulated images, such as those made from projections of N -body models, as long as the units are particles per pixel or something similar.

¹⁰ In practice, the time spent by IMFIT is dominated by the per-pixel model calculations, so any extra time spent re-estimating the per-pixel σ_i values is often negligible.

Fitting a model to data using the Cash statistic is enabled in IMFIT by a command-line flag. The only drawback to using the Cash statistic is that, as explained in Section 4.3, doing so *also* requires using one of the slower minimization techniques. (Fast least-squares minimization techniques assume that the objective function is always positive, which is true for χ^2 but *not* for C .)

4.2. Implementation: Specifying Per-Pixel Errors and Masking

IMFIT’s default behavior, as mentioned above, is to use χ^2 as the statistic for minimization. To do so, the individual, per-pixel Gaussian errors σ_i must be available. If a separate error or noise map is not supplied by the user (see below), IMFIT estimates σ_i values from either the data values or the model values, using the Gaussian approximation to Poisson statistics. To ensure this estimate is as accurate as possible, the data or model values I_i must at some point be converted from counts to actual detected photons (e.g., photoelectrons), and any previously subtracted background must be accounted for.

By default, IMFIT estimates the σ_i values from the data image by including the effects of A/D gain, prior subtraction of a (constant) background, and read noise. Rather than converting the image to electrons pixel⁻¹ and then estimating the σ values, IMFIT generates σ values in the same units as the input image:

$$\sigma_{I,i}^2 = (I_{d,i} + I_{\text{sky}})/g_{\text{eff}} + N_c \sigma_{\text{RN}}^2/g_{\text{eff}}^2, \quad (11)$$

where $I_{d,i}$ is the data intensity in counts pixel⁻¹, I_{sky} is any pre-subtracted sky background in the same units, σ_{RN} is the read noise in electrons, N_c is the number of separate images combined (averaged or median) to form the data image, and g_{eff} is the “effective gain” (the product of the A/D gain, N_c , and optionally the exposure time if the image pixel values are actually in units of counts s⁻¹ pixel⁻¹ rather than integrated counts pixel⁻¹). If model-based χ^2 minimization is used, then model intensity values $I_{m,i}$ are used in place of $I_{d,i}$ in Equation 4.2. In this case, the $\sigma_{I,i}$ values must be recomputed each time a new model image is generated, though in practice this adds very little time to the overall fitting process.

If a mask image has been supplied, it is converted internally so that its pixels have values $z_i = 1$ for valid pixels and $z_i = 0$ for bad pixels. Then the mask values are divided by the variances to form a weight-map image, where individual pixels have values of $w_i = z_i/\sigma_{I,i}^2$. These weights are then used for the actual χ^2 calculation:

$$\chi^2 = \sum_{i=1}^N w_i (I_{d,i} - I_{m,i})^2. \quad (12)$$

Instead of data-based or model-based errors, the user can also supply an error or noise map in the form of a FITS image, such as might be produced by a reduction pipeline. The individual pixel values in this image can be Gaussian errors, variances (σ^2), or even pre-computed weight values w_i .

In the case of Cash-statistic minimization, the sum C is computed directly using Equation 10. The “weight map” is then based directly on the mask image, if any (so all pixels in the resulting weight map have values of $z_i = 0$ or 1). The

actual minimized quantity is thus

$$C = 2 \sum_{i=1}^N z_i (m_i - d_i \ln m_i) \quad (13)$$

with $m_i = g_{\text{eff}}(I_{m,i} + I_{\text{sky}})$ and $d_i = g_{\text{eff}}(I_{d,i} + I_{\text{sky}})$.

4.3. Minimization Algorithms

4.3.1. Levenberg-Marquardt

The default minimization algorithm used by IMFIT is a robust implementation of the Levenberg-Marquardt (L-M) gradient search method (Marquardt 1963), based on the MINPACK-1 version of Moré (1978) and modified by Craig Markwardt (Markwardt 2009),¹¹ which includes optional lower and upper bounds on parameter values. This version of the basic L-M algorithm also includes auxiliary code for doing numerical differentiation of the objective function, and thus the various image functions do not need to provide their own derivatives, which considerably simplifies things when it comes to writing new functions.

The L-M algorithm has the key advantage of being very fast, which is a useful quality when one is fitting large images with a complex set of functions and PSF convolution. It has the minor disadvantage of requiring an initial starting guess for the parameter values, and it has two more significant disadvantages. The first is that like gradient-search methods in general it is prone to becoming trapped in local minima in the objective-function landscape. The second is that it is designed to work with least-squares objective functions, where the objective function values are assumed to be always ≥ 0 . In fact, the L-M algorithm makes use of a vector of the individual contributions from each pixel to the total χ^2 , and these values as well (not just the sum) must be nonnegative. For the χ^2 case, this is always true; but this is *not* guaranteed to be true for the Cash statistic. Thus, it would be quite possible for the L-M minimizer to fail to find the best-fitting solution for a particular image, simply because the solution has a Cash-statistic value < 0 .

4.3.2. Nelder-Mead Simplex

A second, more general algorithm available in IMFIT is the Nelder-Mead simplex method (Nelder & Mead 1965), with constraints as suggested by Box (1965), implemented in the NLOpt library.¹² Like the L-M algorithm, this method requires an initial guess for the parameter set; it also includes optional parameter limits. Unlike the L-M algorithm, it works only with the final objective function value and does not assume that this value must be nonnegative; thus, it is suitable for both χ^2 and Cash-statistic minimization. It is also as a rule less likely to be caught in local minima than the L-M algorithm. The *disadvantage* is that it is considerably *slower* than the L-M method – roughly an order of magnitude so.

4.3.3. Differential Evolution

A third alternative provided by IMFIT is a genetic-algorithms approach called Differential Evolution (DE; Storn & Price 1997). This searches the objective-function landscape using a population of parameter-value vectors; with each “generation”, the population is updated by mutating and

¹¹ <http://purl.com/net/mpfit>

¹² Steven G. Johnson, The NLOpt nonlinear-optimization package, <http://ab-initio.mit.edu/nlopt>.

recombining some of the vectors, with new vectors replacing older vectors if they are better-performing. DE is designed to be – in the context of genetic algorithms – fast and robust while keeping the number of adjustable *algorithm* parameters (e.g., mutation and crossover rates) to a minimum. It is the least likely of the algorithms used by IMFIT to become trapped in a local minimum in the objective-function landscape: rather than starting from a single initial guess for the parameter vector, it begins with a set of randomly generated initial-parameter values, sampled from the full range of allowed parameter values; in addition, the crossover-with-mutation used to generate new parameter vectors for successive generations helps the algorithm avoid local minima traps. Thus, in contrast to the other algorithms, it does not require any initial guesses for the parameter values, but *does* require lower and upper limits for all parameters. It is definitely the *slowest* of the minimization choices: about an order of magnitude slower than the N-M simplex, and thus roughly *two* orders of magnitude slower than the L-M algorithm.

The current implementation of DE in IMFIT uses the “DE/rand-to-best/1/bin” internal strategy, which controls how mutation and crossover are done (Storn & Price 1997), along with a population size of 10 parameter vectors per free parameter. Since the basic DE algorithm has no default stop conditions, IMFIT halts the minimization when the best-fitting value of the fit statistic has ceased to change by more than a specified tolerance after 30 generations, or when a maximum of 600 generations is reached.

4.3.4. Comparison and Recommendations

For most purposes, the default L-M method is probably the best algorithm to use, since it is fast enough to make exploratory fitting (varying the set of functions used, applying different parameter limits, etc.) feasible, and also fast enough to make fitting large numbers of individual objects in a reasonable time possible. If the problem is relatively small (modest image size, few image functions) and the user is concerned about possible local minima, then the N-M simplex or even the DE algorithm can be used. If the problem involves data dominated by Poisson statistics in the low-count regime, where the χ^2 approach may produce biased results (see Section 9), then the L-M algorithm is *not* appropriate and one of the other two should be used.

Table 1 provides a general comparison of the different minimization algorithms, including the time taken for each to find the best fit for a very simple case: a 256×256 -pixel cutout of an SDSS *r*-band image of the galaxy IC 3478, fit with a single Sérsic function and convolved with a 51×51 -pixel PSF image. For this simple case, the N-M approach takes ~ 4 times as long as the L-M method, and the DE algorithm takes ~ 60 times as long. (All three algorithms converged to the same solution, so there was no disadvantage to using the L-M method in this case.)

4.4. Outputs and “Goodness of Fit” Measures

When IMFIT finishes, it outputs the parameters of the best fit (along with possible confidence intervals; see Section 5) to the screen and to a text file; it also prints the final value of the fit-statistic. The best-fitting model image and the residual (data – model) image can optionally be saved to FITS files as well.

For fits which minimize χ^2 , IMFIT also prints the *reduced* χ^2 value, which can be used (with caution) as an indication

of the goodness of the fit. For fits which minimize the Cash statistic, there is no direct equivalent to the reduced χ^2 ; the actual value of the Cash statistic does not have any directly useful meaning by itself. However, both χ^2 and the Cash statistic *can* be used to derive comparative measures of how well *different* models fit the same data.

To this end, IMFIT computes two likelihood-based quantities which can be used to compare different models. The first is the Akaike Information Criterion (AIC, Akaike 1974), which is based on an information-theoretic approach. IMFIT uses the recommended, bias-corrected version of this statistic:

$$\text{AIC}_c = -2\ln \mathcal{L} + 2k + \frac{2k(k+1)}{n-k-1}, \quad (14)$$

where \mathcal{L} is the likelihood value, k is the number of (free) parameters in the model and n is the number of data points. The second quantity is the Bayesian Information Criterion (BIC, Schwarz 1978), which is

$$\text{BIC} = -2\ln \mathcal{L} + k \ln n. \quad (15)$$

When two or more models fit to the same data are compared, the model with the *lowest* AIC (or BIC) is preferred, though a difference ΔAIC or ΔBIC of at least ~ 6 is usually required before one model can be deemed clearly superior (or inferior); see, e.g., Takeuchi (2000) and Liddle (2007) for discussions of AIC and BIC in astronomical contexts, and Burnham & Anderson (2002) for more general background. Needless to say, all models being compared in this manner should be fit using the same statistic – i.e., all fit by minimizing the same type of χ^2 (χ_d^2 or χ_m^2), or all fit by minimizing the Cash statistic.

5. CONFIDENCE INTERVALS FOR FITTED PARAMETERS

In addition to its speed, the Levenberg-Marquardt minimization algorithm has the convenient advantage that it can automatically produce a set of approximate, 1- σ confidence intervals for the fitted parameters as a side product of the minimization process; this comes from inverting the Hessian matrix computed during the minimization process (see, e.g., Section 15.5 of Press et al. 1992).

The other minimization algorithms available in IMFIT do not compute confidence intervals. Although one can, as a workaround, re-run IMFIT using the L-M algorithm on a solution that was found using one of the other algorithms, this will only work if χ^2 minimization is being done. If a fit is done minimizing the Cash statistic, then the L-M algorithm cannot be used (see Section 4.3).

A more general method of estimating confidence intervals is provided by bootstrap resampling (Efron 1979). Each iteration of the resampling process generates a new data image by sampling pixel values, with replacement, from the original data image. (What is actually generated inside IMFIT is a resampled vector of pixel *indices* into the image, excluding those indices corresponding to masked pixels.) The fit is then re-run with the best-fit parameters from the original fit as starting values, using the L-M algorithm for χ^2 minimization cases and the N-M simplex algorithm when Cash-statistic minimization is being done. After n iterations, the combined set of bootstrapped parameter values is used as the distribution of parameter values, from which properly asymmetric 68% confidence intervals are directly determined, along with

Table 1
Comparison of Minimization Algorithms

Algorithm	Initial guess required	Bounds required	Local-minimum vulnerability	χ^2 only	Speed	Timing example
Levenberg-Marquardt (L-M)	Yes	No	High	Yes	Fast	2.2s
Nelder-Mead Simplex (N-M)	Yes	No	Medium	No	Slow	9.1s
Differential Evolution (DE)	No	Yes	Low	No	Very Slow	2m15s

Note. — A comparison of the three nonlinear minimization algorithms available in IMFIT. Column 1: Algorithm name. Column 2: Notes whether an initial guess of parameter values required. Column 3: Notes whether lower and upper bounds on all parameter values are required. Column 4: Vulnerability of the algorithm to becoming trapped in local minima in the χ^2 (or other objective function) landscape. Column 5: Notes whether algorithm is only usable for χ^2 minimization (If “No”, then algorithm can also be used for Cash-statistic minimization.) Column 6: General speed. Column 7: Approximate time taken for fitting a 256×256 pixel SDSS galaxy image (single Sérsic function + PSF convolution), using a MacBook Pro with a quad-core Intel Core i7 2.3 GHz CPU (2011 model).

the standard deviation. (The 68% confidence interval corresponds to $\pm 1\sigma$ if the distribution is close to Gaussian.)

The only drawback of the bootstrap-resampling approach is the cost in time. Since bootstrap resampling should ideally use a minimum of several hundred to one thousand or more iterations, one ends up, in effect, re-running the fit that many times. (Some time is saved by starting each fit with the original best-fit parameter values, since those will almost always be close to the best-fit solution for the resampled data.)

6. IMAGE FUNCTIONS

Image functions are implemented in IMFIT as subclasses of an abstract base class called `FunctionObject`. The rest of the program does not need to know the details of the individual functions, only that they adhere to the `FunctionObject` interface. This makes it relatively simple to add new image functions to the program: write a header file and an implementation file for the new function, add a reference to it in another source file, and recompile the program. Further notes on how to do this are included in the documentation.

This section describes the various default image functions that come with IMFIT. Specifications for the actual parameters (e.g., the order that IMFIT expects to find them in) are included in the documentation, and a summary of all available function names and their corresponding parameter lists can be printed using the `--list-parameters` command-line flag.

6.1. 2D Components

Most image functions, unless otherwise noted, have two “geometric” parameters: the position angle PA in degrees counter-clockwise from the vertical axis of the image¹³ and the ellipticity $\epsilon = 1 - b/a$, where a and b are the semi-major and semi-minor axes, respectively.

In most cases, the image function internally converts the ellipticity to an axis ratio $q = b/a$ ($= 1 - \epsilon$) and the position angle to an angle relative to the image x -axis θ ($= \text{PA} + 90^\circ$), in radians. Then for each input pixel (or subpixel if pixel sub-sampling is being done) with image coordinates (x, y) a scaled radius is computed as

$$r = \left(x_p^2 + \frac{y_p^2}{q^2} \right)^{1/2}, \quad (16)$$

¹³ Reproducing the usual convention for images with standard astronomical orientation, where north is up and east is to the left.

where x_p and y_p are coordinates in the reference frame centered on the image-function center (x_0, y_0) and rotated to its position angle:

$$\begin{aligned} x_p &= (x - x_0) \cos \theta + (y - y_0) \sin \theta \\ y_p &= -(x - x_0) \sin \theta + (y - y_0) \cos \theta \end{aligned} \quad (17)$$

This scaled radius is then used to compute the actual intensity, using the appropriate 1-D intensity function (see descriptions of individual image functions, below).

Pure circular versions of any of these functions can be had by specifying that the ellipticity parameter is fixed, with a value of 0. Some functions (e.g., `EdgeOnDisk`) have only the position angle as a geometric parameter, and instead of computing a scaled radius, convert the pixel coordinates to corresponding r and z values in the rotated 2D coordinate system of the model function.

6.1.1. FlatSky

This is a very basic function which produces a uniform background: $I(x, y) = I_{\text{sky}}$ for all pixels. Unlike most image functions, it has *no* geometric parameters.

6.1.2. Gaussian

This is an elliptical 2D Gaussian function, with central surface brightness I_0 and dispersion σ . The intensity profile is given by

$$I(r) = I_0 \exp\left(-\frac{r^2}{2\sigma^2}\right). \quad (18)$$

6.1.3. Moffat

This is an elliptical 2D function with a Moffat (1969) function for the surface brightness profile, with parameters for the central surface brightness I_0 , full-width half-maximum (FWHM), and the shape parameter β . The intensity profile is given by

$$I(r) = \frac{I_0}{(1 + (r/\alpha)^2)^\beta}, \quad (19)$$

where α is defined as

$$\alpha = \frac{\text{FWHM}}{2\sqrt{2^{1/\beta} - 1}}. \quad (20)$$

In practice, FWHM describes the overall width of the profile, while β describes the strength of the wings: lower values of β

mean more intensity in the wings than is the case for a Gaussian (as $\beta \rightarrow \infty$, the Moffat profile converges to a Gaussian).

The Moffat function is often a good approximation to typical telescope PSFs (see, e.g., Trujillo et al. 2001), and MAKEIMAGE can easily be used to generate Moffat PSF images.

6.1.4. Exponential

This is an elliptical 2D exponential function, with parameters for the central surface brightness I_0 and the exponential scale length h . The intensity profile is given by

$$I(r) = I_0 \exp(-r/h); \quad (21)$$

together with the position angle and ellipticity, there are a total of four parameters. This is a good default for galaxy disks seen at inclinations $\lesssim 80^\circ$, though the majority of disk galaxies have profiles which are more complicated than a simple exponential (e.g., Gutiérrez et al. 2011).

6.1.5. Exponential_GenEllipse

This is the same as the Exponential function, but using generalized ellipses (“boxy” to “disky” shapes) for the isophote shapes. Following Athanassoula et al. (1990) and Peng et al. (2002), the shape of the elliptical isophotes is controlled by the c_0 parameter, such that a generalized ellipse with ellipticity $= 1 - b/a$ is described by

$$\left(\frac{|x|}{a}\right)^{c_0+2} + \left(\frac{|y|}{b}\right)^{c_0+2} = 1, \quad (22)$$

where $|x|$ and $|y|$ are distances from the ellipse center in the coordinate system aligned with the ellipse major axis (c_0 corresponds to $c-2$ in the original formulation of Athanassoula et al.). Thus, values of $c_0 < 0$ correspond to disk isophotes, while values > 0 describe boxy isophotes; $c_0 = 0$ for a perfect ellipse.

6.1.6. Sérsic

This is an elliptical 2D function with the intensity profile given by the Sérsic (1968) function:

$$I(r) = I_e \exp \left\{ -b_n \left[\left(\frac{r}{r_e} \right)^{1/n} - 1 \right] \right\}, \quad (23)$$

where I_e is the surface brightness at the effective (half-light) radius r_e and n is the index controlling the shape of the intensity profile. The value of b_n is formally given by the solution to the transcendental equation

$$\Gamma(2n) = 2\gamma(2n, b_n), \quad (24)$$

where $\Gamma(a)$ is the gamma function and $\gamma(a, x)$ is the incomplete gamma function. However, in the current implementation b_n is calculated via the polynomial approximation of Ciotti & Bertin (1999) when $n > 0.36$ and the approximation of MacArthur et al. (2003) when $n \leq 0.36$.

The Sérsic profile is equivalent to the de Vaucouleurs ($r^{1/4}$) profile when $n = 4$, to an exponential when $n = 1$, and to a Gaussian when $n = 0.5$; it has become the de facto standard for fitting the surface-brightness profiles of elliptical galaxies and bulges.

6.1.7. Sérsic_GenEllipse

This function is similar to the Sérsic function, but uses generalized ellipses for the isophote shapes; see the discussion of the Exponential_GenEllipse function above for details of the isophote shapes. Though the empirical justification for doing so is rather limited, the combination of boxy isophotes and Sérsic profiles with $n < 1$ is often used to represent bars when fitting images of disk galaxies. In addition, the combination of boxy isophotes and high n values may be appropriate for modeling luminous boxy elliptical galaxies.

6.1.8. Core-Sérsic

This function generates an elliptical 2D function where the major-axis intensity profile is given by the Core-Sérsic model (Graham et al. 2003; Trujillo et al. 2004), which was designed to fit the profiles of so-called “core” galaxies (e.g., Ferrarese et al. 2006; Richings et al. 2011; Dullo & Graham 2012, 2013; Rusli et al. 2013). It consists of a Sérsic profile (parameterized by n and r_e) for radii $>$ the break radius r_b and a single power law with index $-\gamma$ for radii $< r_b$. The transition between the two regimes is mediated by the dimensionless parameter α : for low values of α , the transition is very gradual and smooth, while for high values of α the transition becomes very abrupt (a perfectly sharp transition can be approximated by setting α equal to some large number, such as 100). The intensity profile is given by

$$I(r) = I_b \left[1 + \left(\frac{r_b}{r} \right)^\alpha \right]^{\gamma/\alpha} \exp \left[-b \left(\frac{r^\alpha + r_b^\alpha}{r_e^\alpha} \right)^{1/(n\alpha)} \right], \quad (25)$$

where b is the same as b_n for the Sérsic function.

The overall intensity scaling is set by I_b , the intensity at the break radius r_b :

$$I_b = I_0 2^{-\gamma/\alpha} \exp[b 2^{1/\alpha n} (r_b/r_e)^{1/n}]. \quad (26)$$

6.1.9. BrokenExponential

This is similar to the Exponential function, but it has *two* exponential radial zones (with different scalelengths) joined by a transition region at R_b of variable sharpness:

$$I(r) = S I_0 e^{-\frac{r}{h_1}} [1 + e^{\alpha(r-R_b)}]^{-\frac{1}{\alpha}(\frac{1}{h_1} - \frac{1}{h_2})}, \quad (27)$$

where I_0 is the central intensity of the inner exponential, h_1 and h_2 are the inner and outer exponential scale lengths, R_b is the break radius, and α parameterizes the sharpness of the break. Low values of α mean very smooth, gradual breaks, while high values correspond to abrupt transitions. S is a scaling factor,¹⁴ given by

$$S = (1 + e^{-\alpha R_b})^{-\frac{1}{\alpha}(\frac{1}{h_1} - \frac{1}{h_2})}; \quad (28)$$

see Figure 2 for examples. Note that the parameter α has units of length^{-1} (pixels⁻¹ for the specific case of IMFIT).

The 1D form of this profile (Erwin et al. 2008) was designed to fit the surface-brightness profiles of disks which are not single-exponential: e.g., disks with truncations or antitruncations (Erwin et al. 2005; Erwin et al. 2008; Muñoz-Mateos et al. 2013).

¹⁴ As pointed out by Muñoz-Mateos et al. (2013), the original definition of this factor in Eqn. 6 of Erwin et al. (2008) contained a typo.

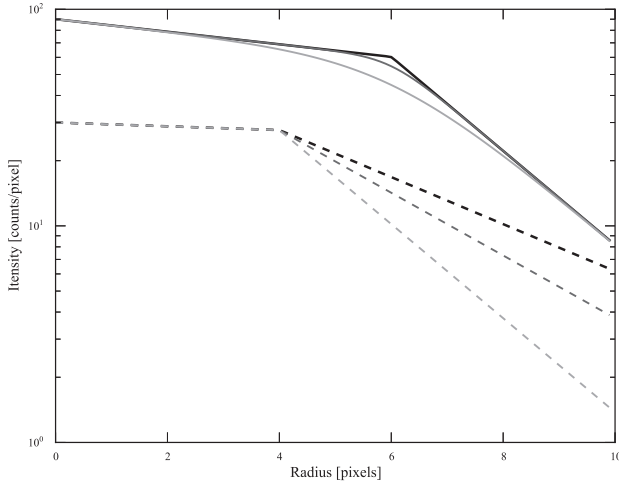


Figure 2. Examples of the “broken-exponential” surface-brightness profile used by the BrokenExp and BrokenExp3D image functions. The upper (solid) curves show a profile with inner and outer scale lengths $h_1 = 15$ and $h_2 = 2$ pixels, respectively, break radius = 6 pixels, and varying values of α (black = 100, medium gray = 3, light gray = 1). The lower (dashed) curves show the effects of varying the outer scale length only ($h_2 = 4, 3, 2$ pixels).

6.1.10. GaussianRing

This function creates an elliptical ring with a Gaussian radial profile, centered at $r = R_{\text{ring}}$ along the major axis.

$$I(r) = I_0 \exp\left(-\frac{(r - R_{\text{ring}})^2}{2\sigma^2}\right). \quad (29)$$

See Figure 3 for an example.

6.1.11. GaussianRing2Side

This function is similar to GaussianRing, except that it uses an asymmetric Gaussian, with different values of σ for $r < R_{\text{ring}}$ and $r > R_{\text{ring}}$. That is, the profile behaves as

$$I(r) = I_0 \exp\left(-\frac{(r - R_{\text{ring}})^2}{2\sigma_{\text{in}}^2}\right) \quad (30)$$

for $r < R_{\text{ring}}$, and

$$I(r) = I_0 \exp\left(-\frac{(r - R_{\text{ring}})^2}{2\sigma_{\text{out}}^2}\right) \quad (31)$$

for $a > R_{\text{ring}}$; see Figure 3 for an example.

6.1.12. EdgeOnDisk

This function provides the analytic form for a perfectly edge-on disk with a radial exponential profile, using the Bessel-function solution of van der Kruit & Searle (1981) for the radial profile. Although it is common to assume that the vertical profile for galactic disks follows a sech^2 function, based on the self-gravitating isothermal sheet model of Spitzer (1942), van der Kruit (1988) suggested a more generalized form for this, one which enables the profile to range from sech^2 at one extreme to exponential at the other:

$$L(z) \propto \text{sech}^{2/n}(nz/(2z_0)), \quad (32)$$

with z the vertical coordinate and z_0 the vertical scale height. The parameter n produces a sech^2 profile when $n = 1$, sech when $n = 2$, and converges to an exponential as $n \rightarrow \infty$. See

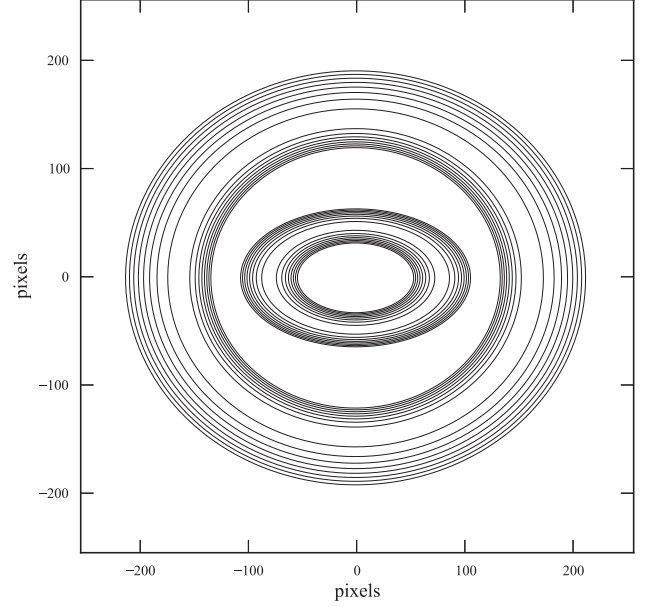


Figure 3. Logarithmically scaled isophotes for examples of the Gaussian ring image functions. The inner, more elliptical ring was generated by the GaussianRing function, with an ellipticity of 0.4, semi-major axis of 80 pixels, and $\sigma = 10$ pixels. The larger, rounder ring is an example of the GaussianRing2Side function, with an ellipticity of 0.1, a semi-major axis of 160 pixels, $\sigma_{\text{in}} = 10$ pixels, and $\sigma_{\text{out}} = 20$ pixels.

de Grijs et al. (1997) for examples of fitting the vertical profiles of edge-on galaxy disks using this formula, and Yoachim & Dalcanton (2006) for examples of 2D fitting of edge-on galaxy images.

In a coordinate system aligned with the edge-on disk, r is the distance from the minor axis (parallel to the major axis) and z is the perpendicular direction, with $z = 0$ on the major axis. (The latter corresponds to height z from the galaxy mid-plane.) The intensity at (r, z) is given by

$$I(r, z) = \mu(0, 0) (r/h) K_1(r/h) \text{sech}^{2/n}(nz/(2z_0)) \quad (33)$$

where h is the exponential scale length in the disk plane, z_0 is the vertical scale height, and K_1 is the modified Bessel function of the second kind. The central surface brightness $\mu(0, 0)$ is given by

$$\mu(0, 0) = 2hL_0, \quad (34)$$

where L_0 is the central luminosity density (see van der Kruit & Searle 1981). Note that L_0 is the actual input parameter required by the function; $\mu(0, 0)$ is calculated internally.

The result is a function with five parameters: L_0 , h , z_0 , n , and the position angle; Figure 4 shows three examples with differing vertical profiles parameterized by $n = 1, 2$, and 100.

6.1.13. EdgeOnRing

This is a simplistic model for an edge-on ring, using two offset subcomponents located at distance R_{ring} from the center of the function block. Each subcomponent (i.e., each side of the ring) is a 2D Gaussian with central surface brightness I_0 and dispersions of σ_r in the radial direction and σ_z in the vertical direction. It has five parameters: I_0 , R_{ring} , σ_r , σ_z , and the position angle. See Figure 5 for examples of this function.

A potentially more correct (though computationally more expensive) model for a ring seen edge-on ring – or at other

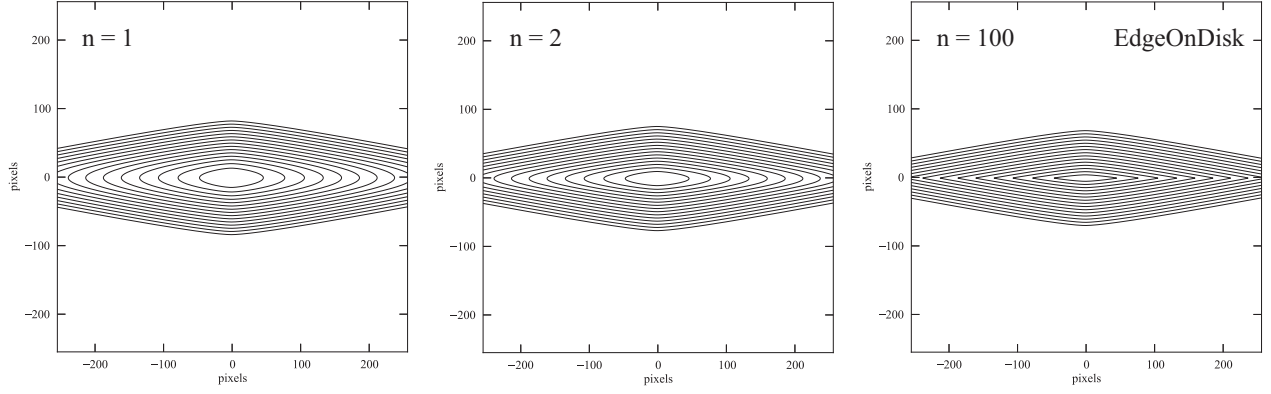


Figure 4. Examples of the EdgeOnDisk image function, which uses the analytic Bessel-function solution of van der Kruit & Searle (1981) for a perfectly edge-on exponential disk, combined with the generalized $\text{sech}^{2/n}$ vertical profile of van der Kruit (1988). All panels show models with radial and vertical scale lengths $h = 50$ and $z_0 = 10$ pixels, respectively. From left to right, the panels show images with vertical $\text{sech}^{2/n}$ profiles having $n = 1$ (sech^2 profile), 2 (sech profile), and 100 (\approx exponential profile).

inclinations – is provided by the GaussianRing3D function, below.

6.1.14. EdgeOnRing2Side

This is a slightly more sophisticated variant of EdgeOnRing, where the radial profile for the two components is an asymmetric Gaussian, as in the case of the GaussianRing2Side function, above: the inner ($|r| < R_{\text{ring}}$) side of each component is a Gaussian with radial dispersion $\sigma_{r,\text{in}}$, while the outer side has radial dispersion $\sigma_{r,\text{out}}$. It thus has six parameters: I_0 , R_{ring} , $\sigma_{r,\text{in}}$, $\sigma_{r,\text{out}}$, σ_z , and the position angle. See the right-hand panel of Figure 5 for an example.

6.2. 3D Components

All image functions in IMFIT produce 2D surface-brightness output. However, there is nothing to prevent one from creating a function which does something quite complicated in order to produce this output. As an example, IMFIT includes three image functions which perform line-of-sight integration through 3D luminosity-density models, in order to produce a 2D projection.

These functions assume a symmetry plane (e.g., the disk plane for a disk galaxy) which is inclined with respect to the line of sight; the inclination is defined as the angle between the line of sight and the normal to the symmetry plane, so that a face-on system has $i = 0^\circ$ and an edge-on system has $i = 90^\circ$. For inclinations $> 0^\circ$, the orientation of the line of nodes (the intersection between the symmetry plane and the sky plane) is specified by a position-angle parameter θ . Instead of a 2D surface-brightness specification (or 1D radial surface-brightness profile), these functions specify a 3D luminosity density j , which is numerically integrated along the line of sight s for each pixel of the model image:

$$I(x, y) = \int_{-S}^S j(s) ds. \quad (35)$$

To carry out the integration for a pixel located at (x, y) in the image plane, the coordinates are first transformed to a rotated image plane system (x_p, y_p) centered on the coordinates of the component center (x_0, y_0) , where the line of nodes lies along the x_p axis (cf. Eqn. 17 in Section 6.1):

$$\begin{aligned} x_p &= (x - x_0) \cos \theta + (y - y_0) \sin \theta \\ y_p &= -(x - x_0) \sin \theta + (y - y_0) \cos \theta \end{aligned}$$

with θ being the angle between the line of nodes and the image $+x$ axis (as in the case of the 2D functions, the actual user-specified parameter is $\text{PA} = \theta - 90^\circ$, which is the angle between the line of nodes and the $+y$ axis).

The line-of-sight coordinate s is then defined so that $s = 0$ in the sky plane (an instance of the image plane located in 3D space so that it passes through the center of the component), corresponding to

$$\begin{aligned} x_{d,0} &= x_p \\ y_{d,0} &= y_p \cos i \\ z_{d,0} &= y_p \sin i \end{aligned}$$

in the component's native (x_d, y_d, z_d) Cartesian coordinate system. A location at s along the line of sight then maps into the component coordinate system as

$$\begin{aligned} y_d &= y_{d,0} + s \sin i \\ z_d &= z_{d,0} - s \cos i, \end{aligned}$$

with $x_d = x_{d,0} = x_p$ by construction. The luminosity-density value is then $j(s) = j(x_d, y_d, z_d)$. See Figure 6 for a side-on view of this arrangement.

Although a fully correct integration would run from $s = -\infty$ to ∞ , in practice the limit S is some large multiple of the component's normal largest scale size (e.g., 20 times the horizontal disk scale length h), to limit the possibility of numerical integration mishaps.

6.2.1. ExponentialDisk3D

This function implements a 3D luminosity density model for an axisymmetric disk where the radial profile of the luminosity density is an exponential and the vertical profile follows the $\text{sech}^{2/n}$ function of van der Kruit (1988) (see the discussion of the EdgeOnDisk function in Section 6.1.12). The line-of-sight integration is done numerically, using functions from the GNU Scientific Library.

In a cylindrical coordinate system (r, z) aligned with the disk (where the disk midplane has $z = 0$), the luminosity density $j(r, z)$ at radius r from the central axis and at height z from the midplane is given by

$$j(r, z) = J_0 \exp(-r/h) \text{sech}^{2/n}(nz/(2z_0)) \quad (36)$$

where h is the exponential scale length in the disk plane, z_0 is the vertical scale height, n controls the shape of the vertical

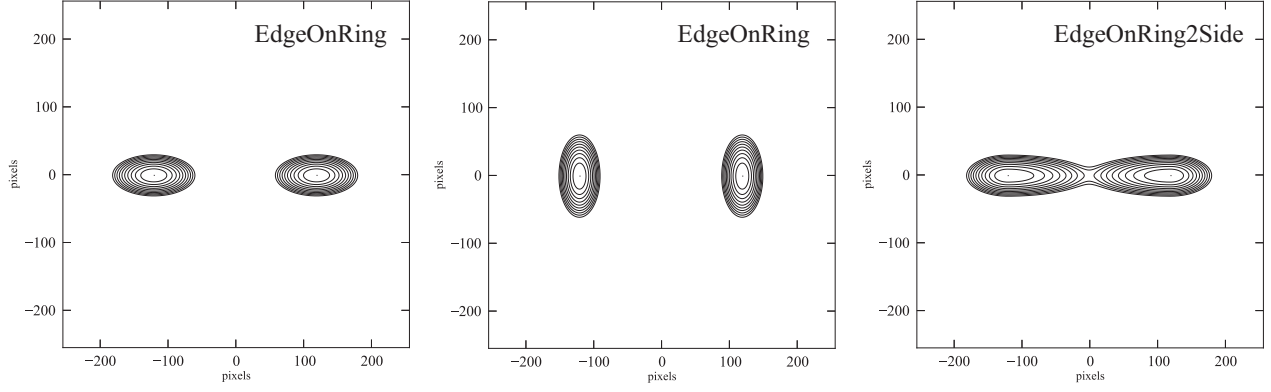


Figure 5. Examples of the EdgeOnRing (left and middle panels) and EdgeOnRing2Side (right panel) image functions, which provide simple approximations for rings seen edge-on. All rings have a radius of 120 pixels. The left-hand panel shows a ring with radial and vertical Gaussian widths of 20 and 10 pixels, respectively; the middle panel shows a model with the radial and vertical widths exchanged. The right-hand panel shows an example of the EdgeOnRing2Side function, where the radial scales are $\sigma = 40$ pixels on the inside and 20 pixels on the outside; the vertical scale is 10 pixels.

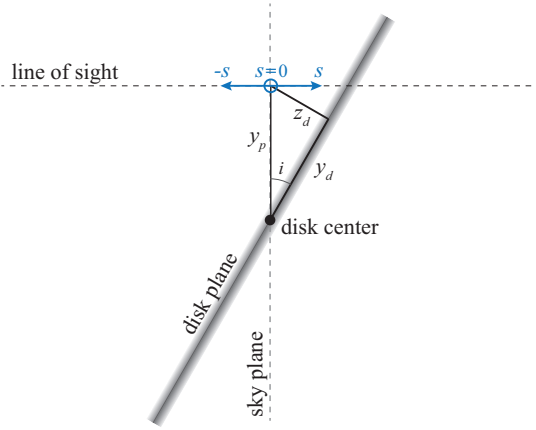


Figure 6. A simplified illustration of how line-of-sight integration is handled for 3D image functions. Here, an axisymmetric ExponentialDisk3D component is inclined at angle i with respect to the line of sight, with the line of nodes rotated to lie along the sky-plane x_p axis, perpendicular to the page; the disk center is by construction at the intersection of the disk plane and the sky plane. For a pixel with sky-plane coordinates (x_p, y_p) , the luminosity-density is integrated along the line of sight (variable s , with $s = 0$ at the sky plane). For each value of s used by the integration routine, the luminosity density is computed based on the corresponding values of radius $r = (x_d^2 + y_d^2)^{1/2}$ and height z_d in the disk's native coordinate system.

distribution, and J_0 is the central luminosity density. Note that in the context of the introductory discussion above, $z = z_d$ and $r = (x_d^2 + y_d^2)^{1/2}$.

Figure 7 shows three views of the same model, at inclinations of 75° , 85° , and 89° ; the latter is almost identical to the image produced by the analytic EdgeOnDisk with the same radial and vertical parameters (right-hand panel of Figure 4).

6.2.2. BrokenExponentialDisk3D

This function is identical to the ExponentialDisk3D function, except that the radial part of the luminosity density function is given by the broken-exponential profile used by the (2D) BrokenExponential function, above (Section 6.1.9). Thus, the luminosity density $j(r, z)$ at radius r from the central axis and at height z from the midplane is given by

$$j(r, z) = I_{\text{rad}}(r) \text{sech}^{2/n}(nz/(2z_0)) \quad (37)$$

where z_0 is the vertical scale height, and the radial part is given by

$$I_{\text{rad}}(r) = SJ_0 e^{-r/h_1} [1 + e^{\alpha(r-R_b)}]^{-\frac{1}{\alpha}(\frac{1}{h_1} - \frac{1}{h_2})}, \quad (38)$$

with J_0 being the central luminosity density and the rest of the parameters as defined for BrokenExponential function (Section 6.1.9).

6.2.3. GaussianRing3D

This function creates the projection of a 3D elliptical ring, seen at an arbitrary inclination. The ring has a luminosity density with a radial Gaussian profile (centered at a_{ring} along ring's major axis, with in-plane width σ) and a vertical exponential profile (with scale height h_z). The ring can be imagined as residing in a plane which has its line of nodes at angle PA and inclination i (as for the ExponentialDisk3D function, above); within this plane, the ring's major axis is at position angle PA_{ring} relative to the perpendicular to the line of nodes. Figure 8 shows the same GaussianRing3D component (with ellipticity = 0.5) seen at three different inclinations.

7. PROGRAMMING NOTES

IMFIT is written in standard C++, and should be compilable with any modern compiler; it has been tested with GCC versions 4.2 and 4.8 on Mac OS X and GCC version 4.6 on Ubuntu Linux systems. It makes use of several open-source libraries, two which are required (CFITSIO and FFTW) and two which are optional but recommended (NLOpt and the GNU Scientific Library). IMFIT also uses the Python-based SCons¹⁵ build system and CxxTest¹⁶ for unit tests.

Since the slowest part of the fitting process is almost always computing the model image, IMFIT is written to take advantage of OpenMP compiler extensions; this allows the computation of the model image to be parceled out into multiple threads, which are then allocated among available processor cores on machines with multiple shared-memory CPUs (including single CPUs with multiple cores). As an example of how effective this can be, tests on a MacBook Pro with a quad-core i7 processor, which has a total of eight virtual threads available, show that basic computation of large images (without PSF convolution) is sped up by a factor of ~ 6 when

¹⁵ <http://www.scons.org>

¹⁶ <http://cxxtest.com>

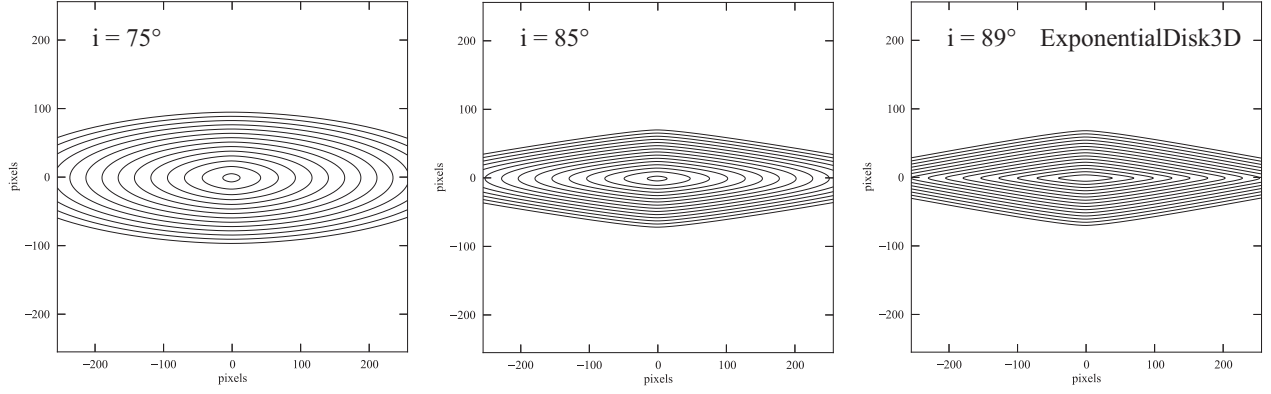


Figure 7. Examples of the ExponentialDisk3D image function, which uses line-of-sight integration through a 3D luminosity-density model of a disk with radial exponential profile and vertical $\text{sech}^{2/n}$ profile. All panels show the same model, with radial and vertical scale lengths $h = 50$ and $z_0 = 10$ pixels, respectively, and a vertical exponential profile ($n = 100$). From left to right, the panels show projections with inclinations of 75° , 85° , and 89° ; compare the last panel with the right-hand panel in Figure 4.

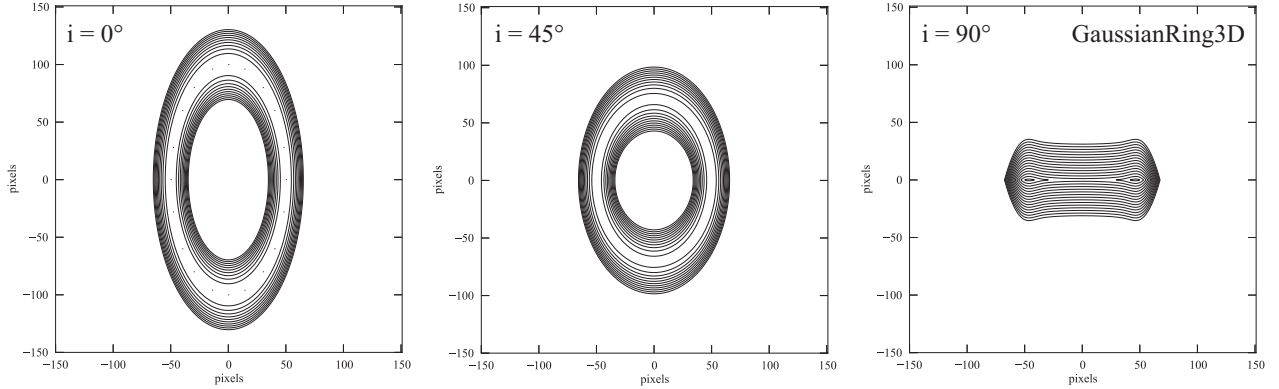


Figure 8. Examples of the GaussianRing3D image function, which uses line-of-sight integration through a 3D luminosity-density model of an elliptical ring with Gaussian radial and exponential vertical profiles. This particular ring has an intrinsic (in-plane) ellipticity = 0.5, semi-major axis = 100 pixels, Gaussian radial width $\sigma = 10$ pixels, and exponential scale height $h_z = 5$ pixels. From left to right, panels show face-on, $i = 45^\circ$, and edge-on views.

OpenMP is used. Even when the overhead of an actual fit is included, the total time to fit a four-component model with 21 free parameters (without PSF convolution) to a 500×500 -pixel image is reduced by a factor of ~ 3.8 .

Additional computational overhead is imposed when one convolves a model image with a PSF. To mitigate this, IMFIT uses the FFTW library to compute the necessary Fourier transforms. This is one of the fastest FFT libraries available, and it can be compiled with support for multiple threads. When the same 500×500 -pixel image fit mentioned above is done including convolution with a 35×35 -pixel PSF image, the total time drops from ~ 280 s without any multi-threading to ~ 120 s when just the FFT computation is multi-threaded, and down to ~ 50 s when OpenMP threading is enabled as well.

Multithreading can always be reduced or turned off using a command-line option if one does not wish to use all available CPU cores for a given fit.

8. SAMPLE APPLICATIONS

IMFIT has been used for several different astronomical applications, including preliminary work on the EUCLID photometric pipeline (Kümmel et al. 2013), testing 1D convolution code used in the analysis of core galaxies (Rusli et al. 2013), fitting kinematically decomposed components of the galaxy NGC 7217 (Fabricius et al. 2014), and separation of bulge and disk components for dynamical modeling of black

hole masses in nearby S0 and spiral galaxies (Erwin et al., in prep).

In this section I present two relatively simple examples of using IMFIT to model images of real galaxies. The first case considers a moderately inclined spiral galaxy with a prominent ring surrounding a bar, where use of a separate ring component considerably improves the fit. The second case is an edge-on disk galaxy with both thin and thick disks; I show how this can be fit using both the analytic pure-edge-on disk component (EdgeOnDisk; Section 6.1.12) and the 3D luminosity-density model of an exponential disk (ExponentialDisk3D; Section 6.2.1).

8.1. PGC 35772: Disk, Bar, and Ring

PGC 35772 is a $z = 0.0287$, early/intermediate-type spiral galaxy (classified as SA0/a by de Vaucouleurs et al. 1993 and as Sb by Fukugita et al. 2007) which was observed as part of the H α Galaxy Groups Imaging Survey (HAGGIS; Kulkarni et al., in prep.) using narrow-band filters on the Wide Field Imager of the ESO 2.2m telescope. The upper-left panel of Figure 9 shows the stellar-continuum-filter image (central wavelength ≈ 659 nm, slightly blueward of the redshifted H α line). Particularly notable is a bright stellar ring, which makes this an interesting test case for including rings in 2D fits of galaxies. Ellipse fits to the image show strong twisting of the isophotal position angle interior to the ring, suggesting a bar

is also present.

The rest of Figure 9 shows the results three different fits to the image, each successive fit adding an extra component. These fits use a 291×281 -pixel cutout of the full WFI image, and were convolved with a Moffat-function image with $\text{FWHM} = 0.98''$, representing the mean PSF (based on Moffat fits to stars in the same image). The best-fit parameters for each model are listed in Table 2.

The first fit is uses a single Sérsic component; the residuals of this fit show a clear excess corresponding to the ring, as well as mis-modeling of the region inside the ring. The fit is improved by switching to an exponential + Sérsic model, with the former component representing the main disk and the latter some combination of the bar + bulge (if any). This two-component model (middle row of the figure) produces less extreme residuals; the best-fitting Sérsic component is significantly elongated and misaligned with the exponential component, so it can be seen to be modeling the bar.

The residuals to the “disk + bar” fit are still significant, however, including the ring itself. To fix this, I include a GaussianRing component (Section 6.1.10) in the third fit (bottom row of Figure 9). The residuals to *this* fit are better not just in the ring region, but also inside, indicating that this three-component model is doing a much better job of modeling the inner flux of the galaxy (the three-component also has the smallest AIC value of the three models; see Table 2).

8.2. IC 5176: Fitting Thin and Thick Disks in an Edge-on Spiral in 2D and 3D

IC 5176 is an edge-on Sbc galaxy, included in a “control” sample of non-boxy-bulge galaxies by Chung & Bureau (2004) and Bureau et al. (2006). Chung & Bureau (2004) noted that both the gas and stellar kinematics were consistent with an axisymmetric, unbarred disk; Bureau et al. (2006) concluded from their *K*-band image that it had a very small bulge and a “completely featureless outer (single) exponential disk.” This suggests an agreeably simple, axisymmetric structure, ideal for an example of modeling an edge-on galaxy. To minimize the effects of the central dust lane (visible in optical images of the galaxy), I use a *Spitzer* IRAC1 ($3.6 \mu\text{m}$) image from S4G (Sheth et al. 2010), retrieved from the *Spitzer* archive. For PSF convolution, I use an in-flight point response function image for the center of the IRAC1 field,¹⁷ down-sampled to the $0.6''$ pixel scale of the post-processed archival galaxy image.

Inspection of major-axis and minor-axis profiles from the IRAC1 image (Figure 10) suggests the presence of both thin and thick disk components; the *K*-band image of Bureau et al. (2006) was probably not deep enough for this to be seen. The major axis profile and the image both suggest a rather round, central excess, consistent with the small bulge identified by Bureau et al.

Consequently, I fit the image using a combination of two exponential-disk models, plus a central Sérsic component for the bulge. The fast way to fit such a galaxy with IMFIT is to assume that the galaxy is perfectly edge-on and use the 2D analytic EdgeOnDisk functions (Section 6.1.12) for the thin and thick disk components. Table 3 shows the results of this fit. The dominant EdgeOnDisk component, which can be thought of as the “thin disk”, has a nearly sech vertical profile and a scale height of $2.0'' \approx 260 \text{ pc}$ (assuming a distance of 26.4 Mpc ; Tully et al. 2009). The second EdgeOnDisk, with a

Table 2
Results of Fitting PGC 35772

Component (1)	Parameter (2)	Value (3)	units (4)
Sersic only (AIC = 12466)			
Sersic	PA	138.7	deg
	ϵ	0.252	
	n	1.021	
	I_e	7.889	counts pix ⁻¹
	r_e	8.256	arcsec
Exponential + Bar (AIC = 11021)			
Exponential (disk)	PA	138.1	deg
	ϵ	0.258	
	I_0	38.85	counts pix ⁻¹
	h	5.18	arcsec
Sersic (bar)	PA	16.1	deg
	ϵ	0.558	
	n	0.948	
	I_e	32.49	counts pix ⁻¹
	r_e	0.745	arcsec
Exponential + Bar + Ring (AIC = 9832)			
Exponential (disk)	PA	140.9	deg
	ϵ	0.276	
	I_0	22.61	counts pix ⁻¹
	h	5.68	arcsec
Sersic (bar)	PA	7.5	deg
	ϵ	0.361	
	n	1.16	
	I_e	16.16	counts pix ⁻¹
	r_e	1.42	arcsec
GaussianRing (ring)	PA	128.1	deg
	ϵ	0.259	
	A	5.46	counts pix ⁻¹
	R	5.54	arcsec
	σ	3.44	arcsec

Results of fitting narrow-band continuum image of spiral galaxy PGC 35771 with progressively more complex models (Sersic; Exponential + Sérsic; Exponential + Sérsic + GaussianRing). “AIC” = Akaike Information Criterion values for the fits; lower values imply better fits.

more exponential-like vertical profile and a scale height of 1.4 kpc , is then the “thick disk” component; it has a radial scale length ~ 2.9 times that of the thin disk.

The central Sérsic component of this model contributes 1.8% of the total flux, while the thin and thick disks account for 70.5% and 27.7%, respectively. The thick/thin-disk luminosity ratio of 0.39 is consistent with the recent study of thick and thin disks by Comerón et al. (2011): using their two assumed sets of relative mass-to-light ratios gives a mass ratio $M_{\text{thick}}/M_{\text{thin}} = 0.47$ or 0.94 , which places IC 5176 in the middle of the distribution for galaxies with similar rotation velocities (see their Fig. 13).

A slower but more general approach is to use the EdgeOnDisk3D function (Section 6.2.1) for both components, which allows for arbitrary inclinations. The cost is in the time taken for the fit: ~ 29 minutes, versus a mere 3m20s for the analytic 2D approach. Using the EdgeOnDisk3D functions *does* give what is formally a better model of the data than the analytic 2D-component fit, with $\Delta\text{AIC} \approx 2305$, though most of the parameter values – in particular, the radial and vertical scale lengths – are almost identical to previous fit. The only notable changes are the Sérsic component becoming rounder (with a different and probably not very well-defined position angle) and the vertical profiles of both disk components becoming pure exponentials (the values of n in Table 3 are im-

¹⁷ <http://irsa.ipac.caltech.edu/data/SPITZER/docs/irac/calibrationfiles/>

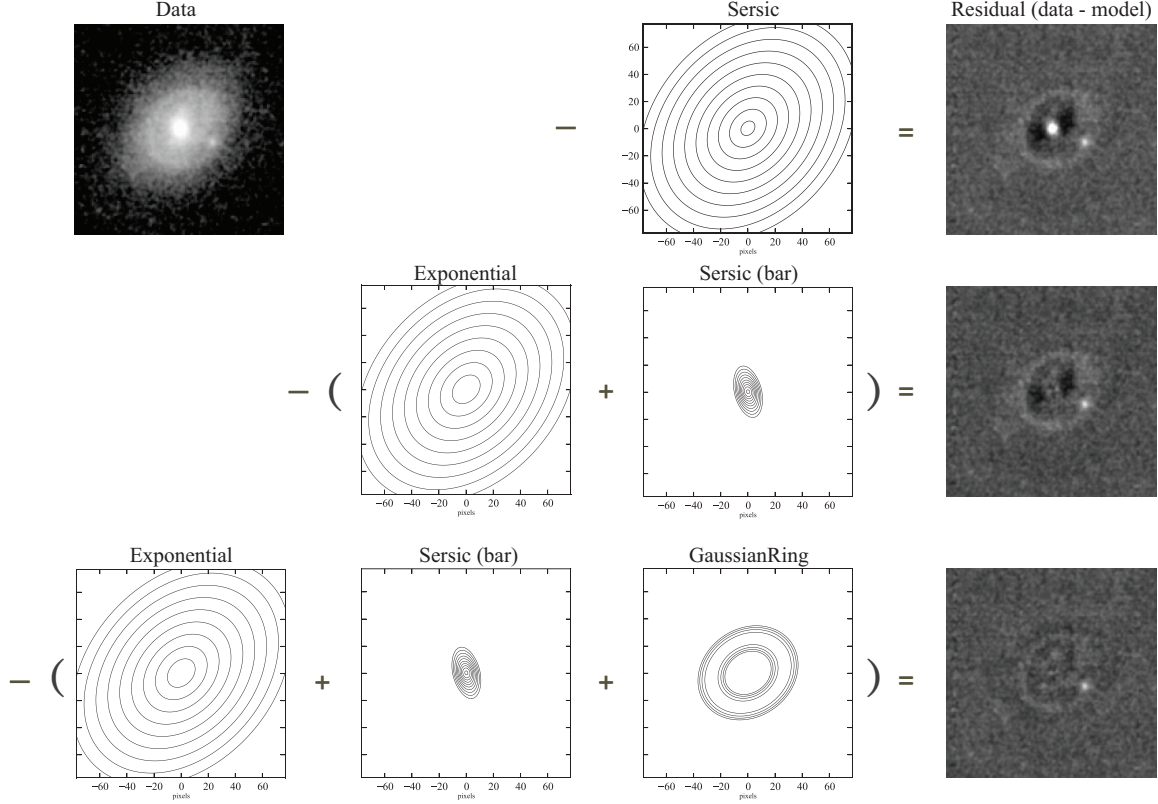


Figure 9. Fits of progressively more complex models to a narrow-band continuum image of the spiral galaxy PGC 35772. Top row: data image, isophote contours of best-fit Sersic model, residual image (data – model) image. Middle row: isophote contours of best-fit Exponential + Sersic components, residual image. Bottom row: isophote contours of best fit Exponential + Sersic + GaussianRing components, residual image.

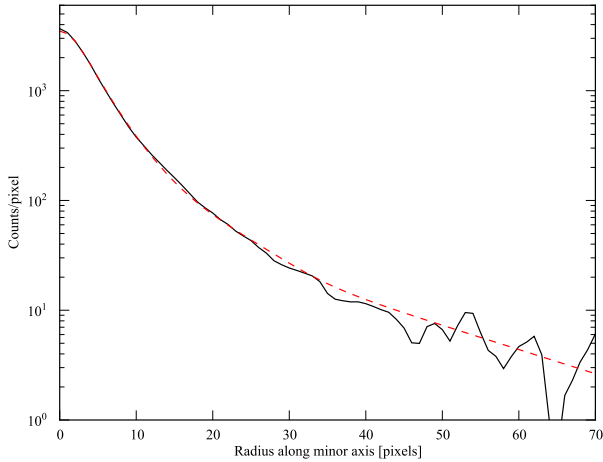


Figure 10. Minor-axis profile from Spitzer IRAC1 image of IC 5176 (black line), along with corresponding profile from best-fitting two-disk model (red dashed line).

posed limits). The relative contributions of the three components are essentially unchanged: 1.8% of the flux from the Sérsic component and 71.7% and 26.5% from the thin and thick disks, respectively. The best-fitting model converges to $i \approx 90^\circ$ for the outer (thick) disk component, but does find $i = 87.2^\circ$ for the thin-disk component.

The reality is that the combination of low spatial resolution

of the IRAC1 image and the presence of residual structure in the disk midplane (probably due to a combination of spiral arms, star formation, and dust) means that we cannot constrain the vertical structure of the disk(s) very well. A vertical profile which is best fit with a sech function when the disk is assumed to be perfectly edge-on can also be fit with a vertical exponential function, if the disk is tilted slightly from edge-on. The low spatial resolution also means that the central bulge is not well constrained, either; the half-light radius of the Sérsic component from either fit is ~ 2.5 pixels and thus barely larger than the seeing.

9. POTENTIAL BIASES IN FITTING GALAXY IMAGES: χ^2 VERSUS CASH-STATISTIC FITS

In Section 4.1, I discussed two different practical approaches to fitting images from a statistical point of view: the standard, Gaussian-based χ^2 statistic and the Poisson-based Cash statistic. The χ^2 approach can be further subdivided into the common method of using data values to estimate the per-pixel errors (χ_d^2) and the alternate method of using values from the model (χ_m^2). Since χ^2 minimization can use very fast algorithms like Levenberg-Marquardt and Cash-statistic minimization cannot, it would seem an obvious choice to use χ^2 , and the only question might be whether to use data-based or model-based error estimation. Even in the case of low S/N, when the Gaussian approximation to Poisson statistics – which motivates the χ^2 approach – might start to become invalid, the presence of Gaussian read noise in CCD detectors might conceivably make this a non-issue. Is there any reason for using the Cash-statistic approach outside of very-

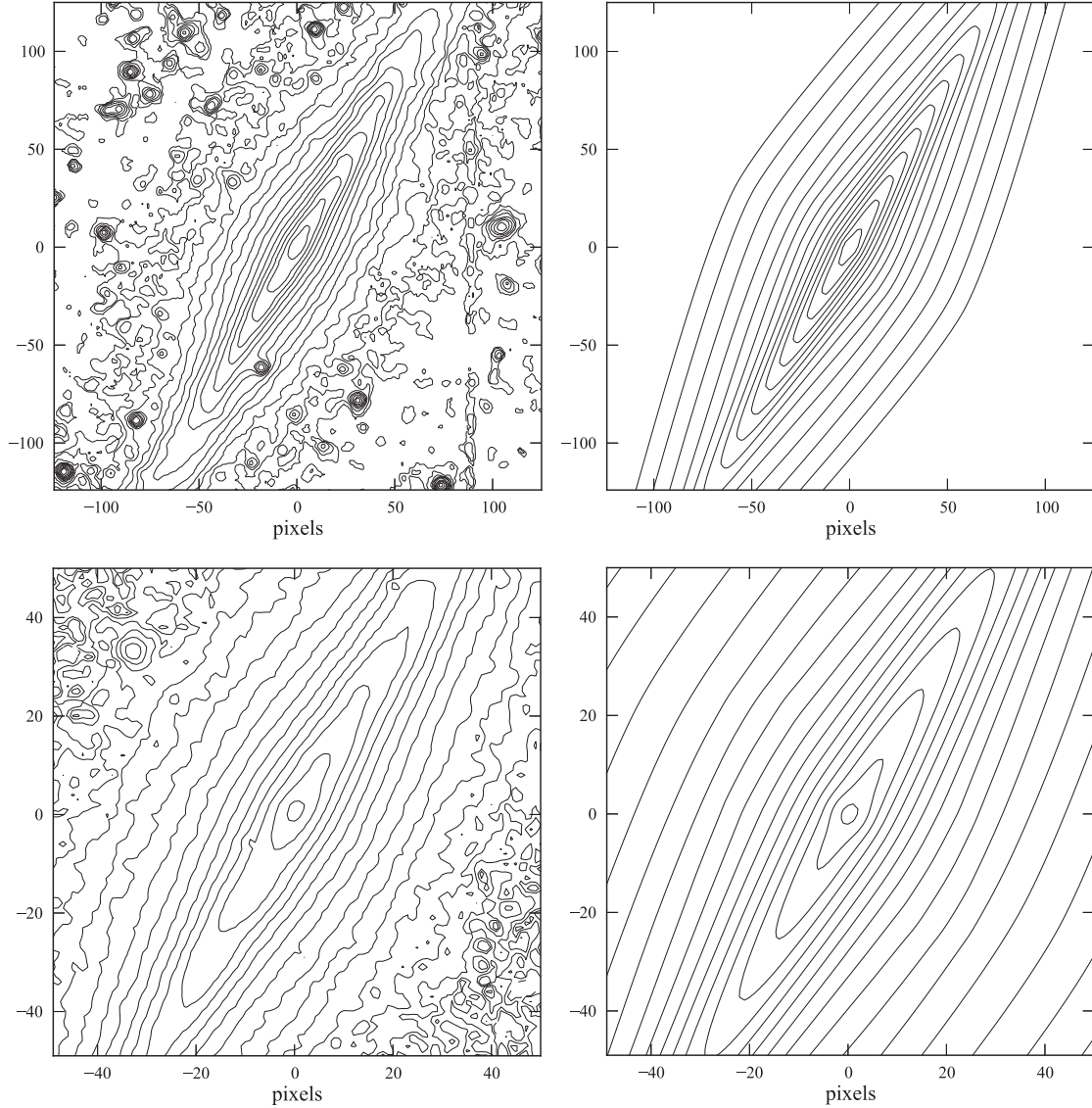


Figure 11. *Left:* Logarithmically scaled isophotes of the Spitzer IRAC1 image of IC 5176. *Right:* Best-fitting, PSF-convolved thin+thick-disk model, using two ExponentialDisk3D components. The upper panels have been smoothed with a 5-pixel-wide median filter.

low-count, zero-read-noise regimes?

Humphrey et al. (2009) used a combination of analytical approximations and fits of models to artificial data to demonstrate that χ^2 fits (using data-based or model-based errors) can lead to biased parameter estimation, even for surprisingly high S/N ratios; these biases were essentially absent when Cash-statistic minimization was used. A fuller discussion of these issues in the context of fitting X-ray data can be found in that paper, and references therein (e.g. Nousek & Shue 1989). In this section, I focus on the typical optical imaging problem of fitting galaxy images with simple 2D functions and use the flexibility of IMFIT to explore how fitting Poisson (or Poisson + Gaussian) data with different assumptions can bias the resulting fitted parameter values.

9.1. Fitting Simple Model Galaxy Images

As a characteristic example, I consider a model galaxy described by a 2D Sérsic function with $n = 3.0$, $r_e = 20$ pixels, and an ellipticity of 0.5. This model is realized in three count-

level regimes: a “low-S/N” case with a sky background level of 20 counts/pixel and model intensity at the half-light radius $I_e = 50$ counts/pixel; a “medium-S/N” version which is equivalent to an exposure time (or telescope aperture) five times larger (background level = 100, $I_e = 250$); and a “high-S/N” version with total counts equal to 25 times the low-S/N version (background level = 500, $I_e = 1250$). These values are chosen partly to test the question of how rapidly the Gaussian approximation to Poisson statistics becomes appropriate: 20 counts/pixel is often given as a reasonable lower limit for using the Gaussian approximation (e.g., Cash 1979), while for 500 counts/pixel the Gaussian approximation should be effectively indistinguishable from true Poisson statistics.

The images were created using code written in Python. The first stage was generating a noiseless 150×150 -pixel reference image (including subpixel integration, but not PSF convolution). This was then used as the source for generating 500 “observed” images of the same size, using Poisson statistics: for each pixel, the value in the reference image was taken as

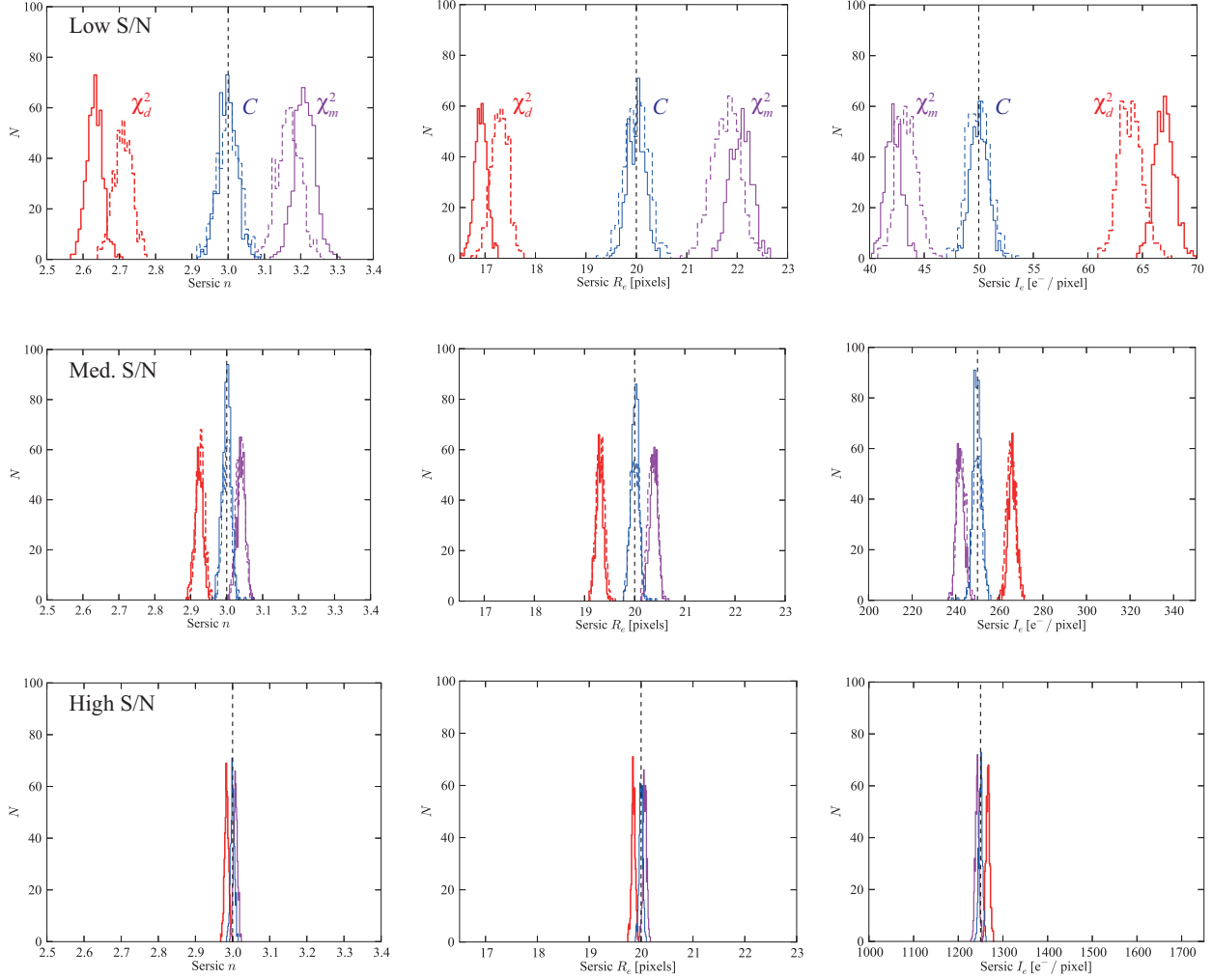


Figure 12. Distribution of best-fit parameters from fits to 500 realizations of an artificial galaxy image with an elliptical Sérsic component + sky background and pure Poisson noise (solid histograms), or Poisson noise + Gaussian read noise ($\sigma = 5 \text{ e}^-$, dashed histograms). Fits used data-based χ^2_d minimization (χ^2_d , red histograms), model-based χ^2_m minimization (χ^2_m , magenta), or Cash-statistic minimization (C, blue); vertical dashed gray lines indicate parameter values of the original model. *Upper row:* Results for low-S/N images (sky background = $20 \text{ e}^-/\text{pixel}$, Sérsic model $I_e = 50 \text{ e}^-/\text{pixel}$). *Middle row:* Results for medium-S/N images (background = $100 \text{ e}^-/\text{pixel}$, $I_e = 250 \text{ e}^-/\text{pixel}$). *Bottom row:* Results for high-S/N images (background = $500 \text{ e}^-/\text{pixel}$, $I_e = 1250 \text{ e}^-/\text{pixel}$); the additional histograms for fits to images with Gaussian read noise are in this case essentially indistinguishable from the pure-Poisson-noise histograms and are not plotted. Using χ^2_d minimization systematically underestimates (Sérsic n and r_e) or overestimates (I_e) the parameters, while using χ^2_m minimization produces smaller biases in the opposite directions. These biases diminish as the counts/pixel get larger. The presence of Gaussian read noise reduces the χ^2 bias in the low-S/N regime, but does not eliminate it. In all cases, the Cash-statistic minimization is bias-free.

the mean m for a Poisson process (Equation 4.1.1), and actual counts were (pseudo)randomly generating using code in the Numpy package (`numpy.random.poisson`).¹⁸ For simplicity, the gain was set to 1, so 1 count = 1 photoelectron.

The resulting images were then fit with IMFIT three times, always using the Nelder-Mead simplex method as the minimization algorithm. The first two fits used χ^2 statistics, either the data-based χ^2_d or the model-based χ^2_m approach, with read noise set to 0; the third fit used the Cash C statistic. The fitted model consisted of a single 2D Sérsic function with very broad parameter limits and the same starting parameter values for all fits (with the initial I_e value scaled by 5 for the medium-S/N images and by 25 for the high-S/N images), along with a fixed FlatSky component for the background.

Figure 12 shows the distribution of best-fit parameters for

fits to all 500 individual images in each S/N regime, with thick red histograms for the χ^2_d fits, thinner magenta histograms for the χ^2_m fits, and thin blue histograms for the Cash-statistic fits, along with the true parameter values of the original model as vertical dashed gray lines.

A clear bias for the χ^2 approaches can be seen in the fits to the low-S/N images (top panels of Figure 12). For the χ^2_d approach, the fitted values of n and r_e are too small: the average value of n is 12.3% low, while the average value of r_e is 15.4% too small. The fitted values of I_e , on the other hand, are on average 34% too large. As can be seen from the figure, these biases are significantly larger than the spread of values from the individual fits. The overall effect also biases the total flux for the Sérsic component, which is underestimated by 10.4% when using the mean parameters of the χ^2_d fit; see Figure 13. The (model-based) χ^2_m approach also produces biases, though these are smaller and are in the opposite sense from

¹⁸ <http://www.numpy.org>

Table 3
Results of Fitting IC 5176

Component (1)	Parameter (2)	Value (3)	units (4)
Fit with 2D Disks (AIC = 182129)			
Sersic (bulge)	PA	149.7	deg
	ϵ	0.206	
	n	0.667	
	I_e	1069.9	counts pix ⁻¹
	r_e	1.48	arcsec
EdgeOnDisk (thin disk)	PA	149.7	deg
	L_0	60.67	counts pix ⁻¹
	h	14.17	arcsec
	n	2.607	
	z_0	2.01	arcsec
EdgeOnDisk (thick disk)	PA	151.3	deg
	L_0	0.677	counts pix ⁻¹
	h	40.97	arcsec
	n	9.89	
	z_0	10.88	arcsec
Fit with 3D Disks (AIC = 179824)			
Sersic (bulge)	PA	168.7	deg
	ϵ	0.046	
	n	0.762	
	I_e	886.4	counts pixel ⁻¹
	r_e	1.46	arcsec
EdgeOnDisk3D (thin disk)	PA	149.7	deg
	i	87.2	deg
	L_0	83.17	counts pixel ⁻¹
	h	14.44	arcsec
	n	50	
EdgeOnDisk3D (thick disk)	z_0	2.04	arcsec
	PA	151.4	deg
	i	89.4	deg
	L_0	0.632	counts pixel ⁻¹
	h	42.07	arcsec
	n	50	
	z_0	11.74	arcsec

Results of fitting Spitzer IRAC1 image of the edge-on spiral IC 5176. The first fit uses analytic 2D EdgeOnDisk components (exponential disk seen at $i = 90^\circ$); the second fit uses line-of-sight integration through ExponentialDisk3D components (3D luminosity-density models), for which the inclination i is a free parameter. Size parameters have been converted from pixels to arc seconds. “AIC” = Akaike Information Criterion values for the two fits.

the χ_d^2 biases: n and r_e are overestimated on average by 7.0% and 10.4%, respectively, while I_e is 15.6% too small; the total flux is overestimated by 6.5%. Finally, the fits using the Cash statistic are *unbiased*: the histograms straddle the input model values, and the mean values from the fits are all $< 0.1\%$ different from the true values. The other parameters of the fits – galaxy center, position angle, ellipticity – do not show any systematic differences, except for a very slight tendency of the ellipticity to be biased high with the χ_d^2 fit, but only at the $\sim 0.5\%$ level. For the parameters which show biases in the χ^2 fits, the trends are exactly as suggested by Humphrey et al. (2009), including the fact that the χ_m^2 biases are smaller and have the opposite sign from the χ_d^2 biases.

In the medium-S/N case (middle panels of the same figure), the bias in the χ_d^2 and χ_m^2 fits is clearly reduced: for the χ_d^2 fits, n and r_e are on average only 2.6% and 3.5% too small, while I_e is on average 6.4% too high (in the χ_m^2 fits, the deviations are 1.3% and 1.9% too large and 3.2% too small, respectively) – though the bias is clearly still present, and in the same pattern. The biases in total flux are smaller, too: 2.3% low and 1.2%

high for the data-based and model-based χ^2 fits, respectively (Figure 13). These biases are even smaller in the high-S/N case: e.g., in the χ_d^2 case, n and r_e are 0.54% and 0.74% too small, while I_e is 1.3% too high. In both S/N regimes, the Cash-statistic fits remain unbiased.

What is the effect of adding (Gaussian) read noise to the images? To investigate this, additional sets of images were prepared as before, except that the value from the Poisson process was further modulated by adding a Gaussian with mean = 0 and width $\sigma = 5 \text{ e}^{-1}$. (This value was chosen as a representative read noise for typical modern CCDs; it is also roughly equal to the dispersion of the Gaussian approximation to the Poisson noise of the background in the low-S/N limit – i.e., $\sigma_{\text{sky}} \approx \sqrt{20}$.)

The fits were done as before, with the read noise properly included in the χ^2 fitting; the histograms of the resulting fits are shown in Figures 12 and 13 with dashed lines. What is clear from the figure is that while the addition of a Gaussian noise term reduces the bias in the χ^2 fits slightly in the low-S/N regime, the bias is still very much present. Even though the Cash statistic is no longer formally correct when Gaussian noise is present, the Cash-statistic fits remain unbiased in the presence of moderate read noise.

9.2. Quantifying the Bias

How large is the bias produced by χ^2 fits? Humphrey et al. (2009) suggested that the absolute or relative size of the bias might not be as important as the size of the bias relative to the nominal statistical errors of the fits. There are, in principle, three different ways of estimating these errors: from the distribution of the fitted values for all 500 images (similar to what was done by Humphrey et al. for their examples); from the mean of individual-fit error estimates produced by using the L-M algorithm; and from the mean of individual-fit error estimates produced by bootstrap resampling. For this simple model, all three approaches produce very similar values. For example, fitting the images in χ_d^2 mode with the L-M algorithm produces estimated dispersions within $\sim 10\%$ of the dispersion of values from the individual χ_d^2 fits; the latter are in turn very similar to the dispersion of the individual Cash-statistic fits (as is evident from the similar histogram widths in Figure 12). The errors estimated from bootstrap resampling also agree to within $\sim 10\%$ of the other estimates (except for errors on n , which are roughly twice as large in the bootstrap case for the low- and medium-S/N images).

Figure 14 shows the biases for the χ_d^2 , χ_m^2 , and Cash-statistic fits, plotted against the background value for the different S/N regimes: the top panels show the deviations relative to the true parameter values, while the bottom panels show the deviations in units of the statistical errors (using the standard deviation of the 500 fitted values). The left and right panels show the cases for χ_d^2 and χ_m^2 fits, respectively, with the Cash-statistic fits shown in each panel for reference. In all cases, there is a clear trend of the χ^2 biases becoming smaller as the overall exposure level (represented by the mean background level) increases, asymptotically approaching the zero-bias case exhibited by the Cash-statistic fits.

Humphrey et al. (2009) derived an estimate for the bias (relative to the statistical error) that would result from fitting pure-Poisson data using the χ_d^2 statistic, based on the total number of counts N_c and the total number of bins N_{bins} (i.e., the total

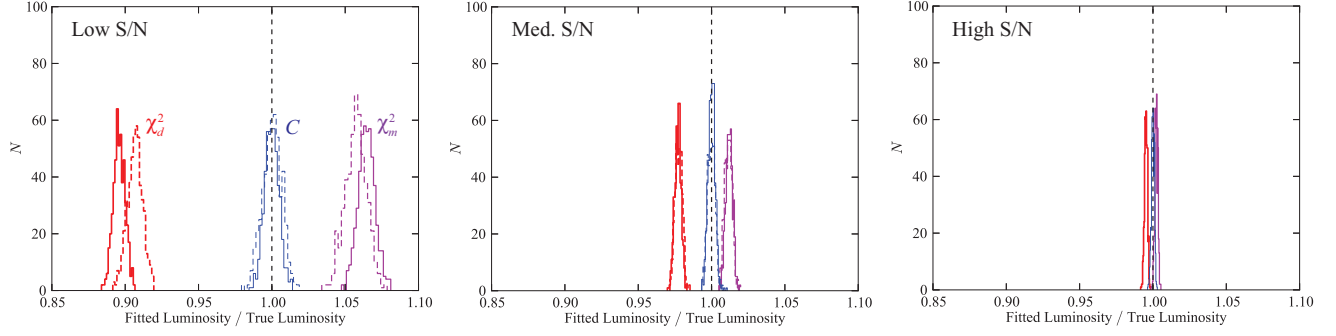


Figure 13. As for Figure 12, but now showing the distribution of the estimated (Sérsic) galaxy luminosity (relative to the true luminosity) from the fits to the model images. In the low-S/N case (left panel), the data-based χ^2_d fits (red) underestimate the true luminosity by 10.4% (9.3% when read noise is present, dashed red histogram), while the model-based χ^2_m fits (magenta) overestimate it by 6.5% (5.8% for the read-noise case); the Cash-statistic fits (blue) are unbiased. These biases diminish in the medium-S/N regime (middle panel) and high-S/N regime (right panel).

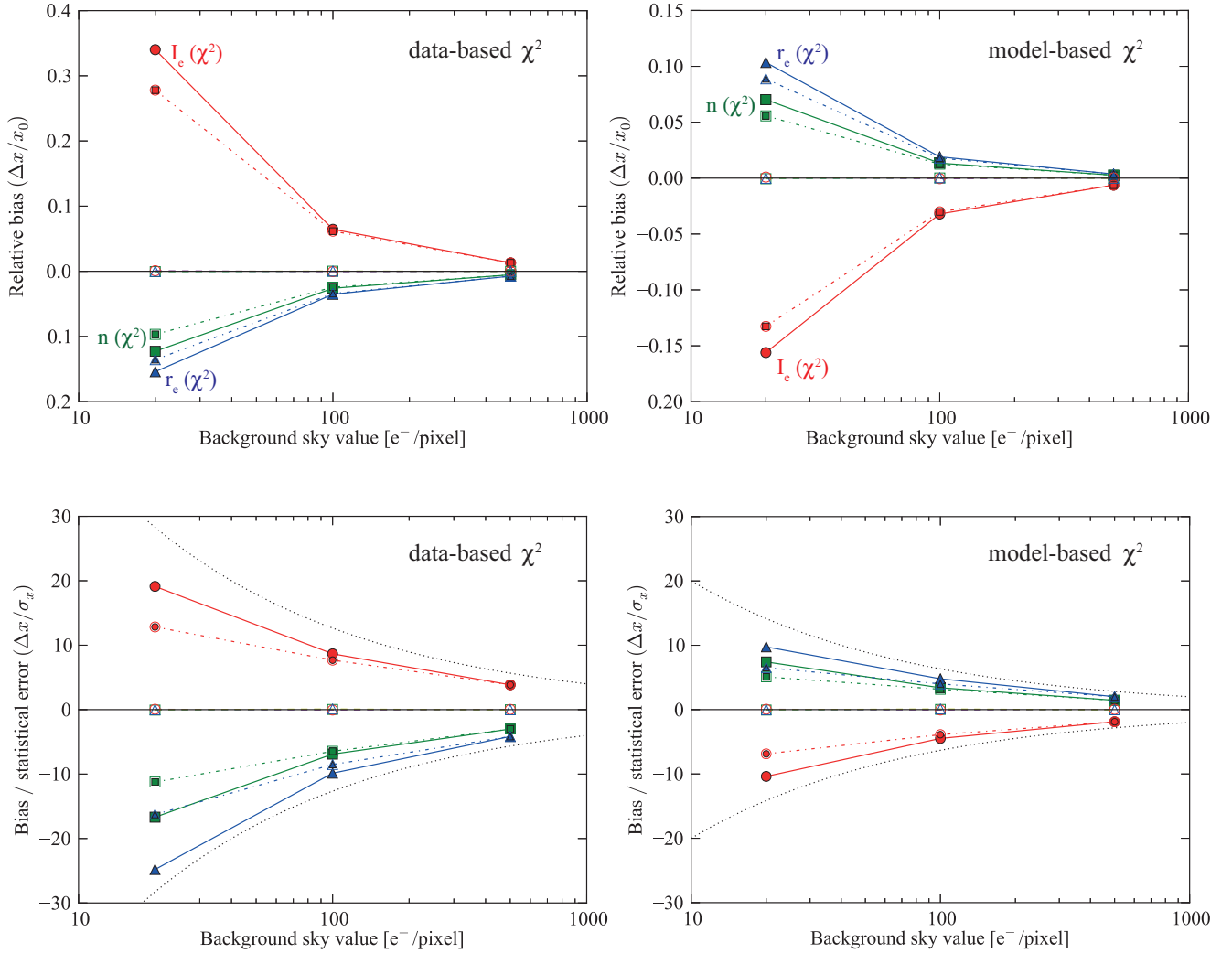


Figure 14. Bias in fitted Sérsic parameters, for the low-, medium-, and high-S/N model images (see Figure 12). The bias is plotted versus the background level of the corresponding images. Solid symbols and lines are for parameter values from χ^2 fits to pure-Poisson images (green = n , red = I_e , blue = r_e), while semi-filled symbols and dot-dashed lines are from χ^2 fits to images with added read noise. Hollow symbols and dashed lines are from Cash-statistic fits, which show essentially no bias. *Top panels:* Fractional bias $(\bar{x} - x_0)/x_0$, where \bar{x} is the mean measured parameter value from fits to 500 images and x_0 is the original model value; the left and right panels show χ^2_d and χ^2_m fits, respectively. *Bottom panels:* Same, but now showing bias relative to statistical error $(\bar{x} - x_0)/\sigma_x$, where σ_x is the nominal statistical error from the fit. The upper and lower dotted curves in each panel show the predicted bias from Humphrey et al. (2009), which is based purely on total counts and number of pixels in the images (Equations 39 and 40).

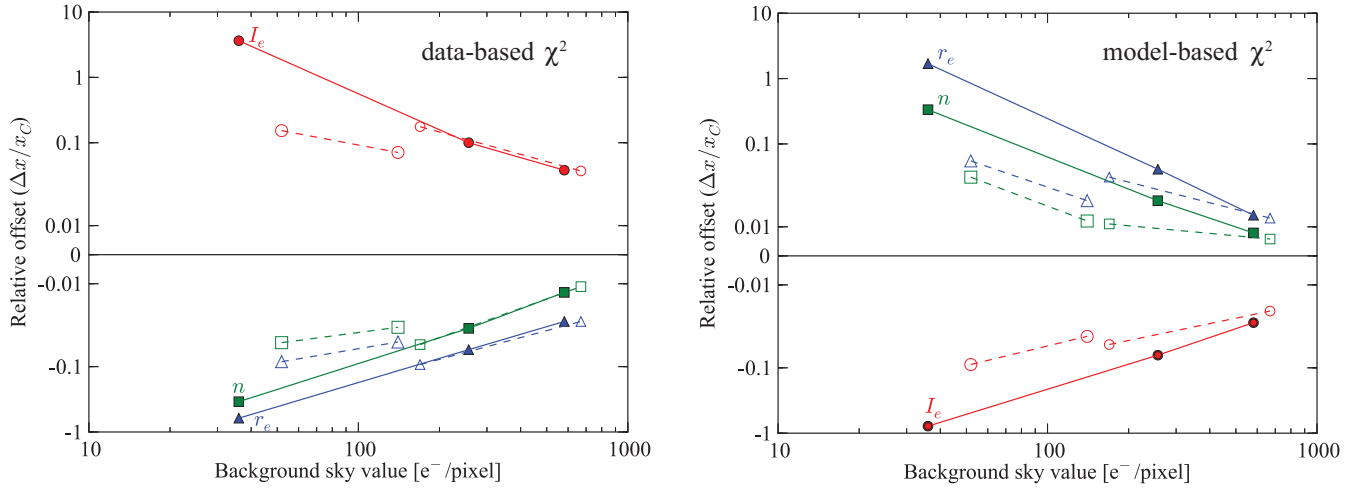


Figure 15. As for the top panels of Figure 14, but now showing relative differences in fitted Sérsic parameters between χ^2 fits and Cash-statistic fits to images of three real elliptical galaxies. Shown is $(x - x_C)/x_C$, where x is the n , r_e , or I_e value from a χ^2 fit and x_C is the value from a Cash-statistic fit to the same image, plotted against mean sky background for the image. *Left panel:* Data-based χ^2 fits. *Right panel:* Model-based χ^2 fits. Solid points are from fits to (in order of increasing background level) SDSS u , g , and r images of NGC 5831; small hollow symbols are from fits to 15s and 60s INT-WFC r -band images of NGC 4697, while larger hollow symbols are from fits to 15s and 40s INT-PFCU V -band images of NGC 3379. Green squares, blue triangles, and red circles indicate Sérsic n , r_e , and I_e , respectively; solid lines connect results for the same galaxy with different exposure levels.

number of fitted pixels):

$$f_b(\chi_d^2) = \frac{|x_0 - \bar{x}|}{\sigma_x} \sim \frac{N_{\text{bins}}}{\sqrt{N_c}}, \quad (39)$$

where x_0 is the true value, \bar{x} is the mean fitted value, and σ_x is the statistical error on the parameter value. They did the same for the χ_m^2 approach and found

$$f_b(\chi_m^2) \sim -0.5 \frac{N_{\text{bins}}}{\sqrt{N_c}}. \quad (40)$$

The estimates derived from these equations are plotted as dotted lines in Figure 14. Although the actual biases are systematically smaller than the predictions, the overall agreement is rather good.

9.3. Biases in Fitting Images of Real Galaxies

Is there any evidence that the χ^2 -bias effect is significant when fitting images of real galaxies? Figure 15 shows the differences seen when fitting single Sérsic functions to images of three elliptical galaxies. In the first case, I fit 996×1121 -pixel cutouts from SDSS u , g , and r images of NGC 5831; these images correspond to successively higher counts per pixel in both background and galaxy. (The cutouts, as well as the mask images, were shifted in x and y to correct for pointing offsets between the different images.) Although color gradients may produce (genuinely) different fits for the different images, these should be small for an early-type galaxy like NGC 5831; more importantly, the bias estimates I calculate (see next paragraph) are between the χ^2 and Cash-statistic fits for each individual band. In the second and third cases I fit same-filter images with different exposure times: 1801×1701 -pixel cutouts from short (15s) and long (60s) r -band exposures of NGC 4697, obtained with the Isaac Newton Telescope’s Wide Field Camera on 2004 March 17, and 15s and 40s V -band exposures of NGC 3379, obtained with the Prime Focus Camera Unit of the INT on 1994 March 14 (image size = 1243×1152 pixels). All images were fit with a single Sérsic function, convolved with an appropriate Moffat PSF image (based on measurements of unsaturated stars in each image). All fits were done with χ_d^2 , χ_m^2 , and Cash-statistic minimization; the χ^2 fits included appropriate read-noise contributions (5.8 e^- and 4.5 e^- for the WFC and PFCU images, respectively).

Unlike the case for the model images in the preceding section, the “correct” Sérsic model for these galaxies is unknown. Thus, Figure 15 shows the differences between the best χ^2 -fit parameters and the parameters from the Cash-statistic fits, relative to the value of the latter, instead of the difference between all three and the (unknown) “true” solution. The trends are nonetheless very similar to the model-image case (compare Figure 15 with the top panels of Figure 14): values of n and r_e from the χ_d^2 fits are smaller, and values of I_e are larger, than the corresponding values from the Cash-statistic fits, and the offsets are reversed when χ_m^2 fitting is done. Although there is some scatter, the tendency of χ_d^2 offsets to be larger than χ_m^2 offsets is present as well: in fact, the average ratio of the former to the latter is 1.99 (median = 1.65), which is strikingly close to the ratio of 2 predicted by Humphrey et al. (2009). Even the fact that the r_e offsets are always larger than the n offsets replicates the pattern from the fits to artificial-galaxy images. In addition, the offsets between the χ^2 -fit values and the Cash-statistic fits diminish as the count rate in-

creases, as in the model-image case. If we make the plausible assumption that the higher S/N images are more likely to yield accurate estimates of the true galaxy parameters (to the extent that the galaxies *can* be approximated by a simple elliptical Sérsic function), then the convergence of estimated parameter values in the high-count regime strongly suggests that the Cash-statistic approach is the least biased in all regimes.

Of course, for many typical optical and near-IR imaging situations, the count rates even in the sky background are high enough that differences between χ^2 and Cash-statistic fits can probably be ignored. For example, typical backgrounds in SDSS g , r , i , and z images range between ~ 60 and 200 ADU/pixel, or ~ 300 – 1000 photoelectrons/pixel.¹⁹ Only for u -band images does the background level become low enough (~ 30 – 150 photoelectrons/pixel) for the χ^2 -fit bias to become a significant issue.

The model-based and data-based χ^2 fits yield (biased) parameters which straddle the parameters from Cash-statistic fits. This suggests that in cases where a Cash-statistic fit would be too time-consuming (because the fast Levenberg-Marquardt algorithm cannot be used), a relatively cheap way of evaluating the possible biases in χ^2 fits is to simply fit an image using both χ_d^2 and χ_m^2 approaches and compare the resulting parameters.

10. SUMMARY

I have described a new open-source program, IMFIT, intended for modeling images of galaxies or other astronomical objects. Key features include speed, a flexible user interface, multiple options for handling the fitting process, and the ability to easily add new 2D image functions for modeling galaxy components.

Images are modeled as the sum of one or more 2D image functions, which can be grouped into multiple sets of functions, each set sharing a common location within the image. Available image functions include standard 2D functions used to model galaxies and other objects – e.g., Gaussian, Moffat, exponential, and Sérsic profiles with elliptical isophotes – as well as broken exponentials, analytic edge-on disks, Core-Sérsic profiles, and symmetric (and asymmetric) rings with Gaussian radial profiles. In addition, several sample “3D” functions compute line-of-sight integrations through 3D luminosity-density models, such as an axisymmetric disk with a radial exponential profile and a vertical $\text{sech}^{2/n}$ profile. Optional convolution with a PSF is accomplished via Fast Fourier Transforms, using a user-supplied FITS image for the PSF.

Image fitting can be done by minimization of the standard χ^2 statistic, using either the image data to estimate the per-pixel variances (χ_d^2) or the computed model values (χ_m^2), or by using user-supplied variance or error maps. Fitting can *also* be done using the Poisson-based Cash statistic instead, which is especially appropriate for cases of images with low counts per pixel and low or zero read noise. In addition, fitting using the Cash statistic does not suffer from biases that afflict χ^2 fits (Humphrey et al. 2009). The only drawback to the Cash statistic is that it cannot be used with the fastest (Levenberg-Marquardt) minimization algorithm, so the somewhat slower Nelder-Mead simplex method (or the even slower Differential Evolution algorithm) must be used instead. Confidence

¹⁹ Based on measurements of ~ 25 SDSS DR7 fields.

intervals for fitted parameters can be estimated by bootstrap resampling.

A comparison of fits to artificial images of a simple Sérsic-function galaxy demonstrates how the χ^2 -bias discussed by Humphrey et al. (2009) manifests itself when fitting images: fits which minimize χ_d^2 result in values of the Sérsic parameters n and r_e (as well as the total luminosity) which are biased low and values of I_e which are biased high, while fits which minimize χ_m^2 produce smaller biases in the opposite directions; as predicted, these biases decrease, but do not vanish, when the background and source intensity levels increase. Fits using the Cash statistic yield essentially *unbiased* parameter values; this is true even when Gaussian read noise is present. Sérsic fits to images of real elliptical galaxies with varying exposure times or background levels show evidence for the same pattern of biased parameter values when minimizing χ_d^2 or χ_m^2 . This suggests that the fitting of galaxy images should use Cash-statistic minimization instead of χ^2 minimization whenever possible, unless time constraints are crucial; even then, Cash-statistic minimization should probably be used unless the background level is at least ~ 100 photoelectrons/pixel.

Precompiled binaries, documentation, and full source code (released under the GNU Public License) are available at the following web site:
<http://www.mpe.mpg.de/~erwin/code/imfit/>.

Various useful comments and suggestions have come from Maximilian Fabricius, Martin Kümmel, and Roberto Saglia, and thanks are also due to Michael Opitsch and Michael Williams for being (partly unwitting) beta testers. Further bug reports, suggestions, requests, and fixes from Giulia Savorgnan, Guillermo Barro, Sergio Pascual, and (especially) André Luiz de Amorim are gratefully acknowledged. This work was partly supported by the Deutsche Forschungsgemeinschaft through Priority Programme 1177, “Witnesses of Cosmic History: Formation and evolution of galaxies, black holes, and their environment.”

This work is based in part on observations made with the *Spitzer* Space Telescope, obtained from the NASA/IPAC Infrared Science Archive, both of which are operated by the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration. This paper also makes use of data obtained from the Isaac Newton Group Archive which is maintained as part of the CASU Astronomical Data Centre at the Institute of Astronomy, Cambridge.

Funding for the creation and distribution of the SDSS Archive has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Aeronautics and Space Administration, the National Science Foundation, the U.S. Department of Energy, the Japanese Monbukagakusho, and the Max Planck Society. The SDSS Web site is <http://www.sdss.org/>.

The SDSS is managed by the Astrophysical Research Consortium (ARC) for the Participating Institutions. The Participating Institutions are The University of Chicago, Fermilab, the Institute for Advanced Study, the Japan Participation Group, The Johns Hopkins University, the Korean Scientist Group, Los Alamos National Laboratory, the Max-Planck-Institute for Astronomy (MPIA), the Max-Planck-Institute for Astrophysics (MPA), New Mexico State University, University of Pittsburgh, University of Portsmouth, Princeton Uni-

versity, the United States Naval Observatory, and the University of Washington.

REFERENCES

- Akaike, H. 1974, *IEEE T. Automat. Contr.*, 19, 716
 Athanassoula, E., Morin, S., Wozniak, H., Puy, D., Pierce, M. J., Lombard, J., & Bosma, A. 1990, *MNRAS*, 245, 130
 Box, M. J. 1965, *The Computer Journal*, 8, 42
 Bureau, M., Aronica, G., Athanassoula, E., Dettmar, R.-J., Bosma, A., & Freeman, K. C. 2006, *MNRAS*, 370, 753
 Burnham, K. P. & Anderson, D. E. 2002, *Model Selection and Multimodal Inference: A Practical Information-Theoretic Approach*, 2nd Ed. (New York: Springer-Verlag)
 Byun, Y. I. & Freeman, K. C. 1995, *ApJ*, 448, 563
 Capaccioli, M., Held, E. V., & Nieto, J.-L. 1987, *AJ*, 94, 1519
 Cappellari, M. 2002, *MNRAS*, 333, 400
 Cash, W. 1979, *ApJ*, 228, 939
 Chung, A. & Bureau, M. 2004, *AJ*, 127, 3192
 Ciotti, L. & Bertin, G. 1999, *A&A*, 352, 447
 Comerón, S., Elmegreen, B. G., Knapen, J. H., Salo, H., Laurikainen, E., Laine, J., Athanassoula, E., Bosma, A., Sheth, K., Regan, M. W., Hinz, J. L., Gil de Paz, A., Menéndez-Delmestre, K., Mizusawa, T., Muñoz-Mateos, J.-C., Seibert, M., Kim, T., Elmegreen, D. M., Gadotti, D. A., Ho, L. C., Holwerda, B. W., Lappalainen, J., Schinnerer, E., & Skibba, R. 2011, *ApJ*, 741, 28
 de Grijs, R., Peletier, R. F., & van der Kruit, P. C. 1997, *A&A*, 327, 966
 de Jong, R. S. 1996, *A&AS*, 118, 557
 de Souza, R. E., Gadotti, D. A., & dos Anjos, S. 2004, *ApJS*, 153, 411
 de Vaucouleurs, G., de Vaucouleurs, A., Corwin, H. G., Buta, R. J., Paturel, G., & Fouqué, P. 1993, *Third Reference Catalog of Bright Galaxies* (New York: Springer-Verlag)
 Dullo, B. T. & Graham, A. W. 2012, *ApJ*, 755, 163
 —. 2013, *ApJ*, 768, 36
 Efron, B. 1979, *Ann.Statist.*, 7, 1
 Emsellem, E., Monnet, G., & Bacon, R. 1994, *A&A*, 285, 723
 Erwin, P., Beckman, J. E., & Pohlen, M. 2005, *ApJL*, 626, L81
 Erwin, P., Pohlen, M., & Beckman, J. E. 2008, *AJ*, 135, 20
 Fabricius, M. H., Cocato, L., Bender, R., Drory, N., Gössel, C., Landriau, M., Saglia, R. P., Thomas, J., & Williams, M. J. 2014, *MNRAS*, 441, 2212
 Ferrarese, L., Côté, P., Jordán, A., Peng, E. W., Blakeslee, J. P., Piatek, S., Mei, S., Merritt, D., Milosavljević, M., Tonry, J. L., & West, M. J. 2006, *ApJS*, 164, 334
 Freeman, P., Doe, S., & Siemiginowska, A. 2001, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Vol. 4477, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ed. J.-L. Starck & F. D. Murtagh, 76–87
 Frigo, M. & Johnson, S. G. 2005, *Proceedings of the IEEE*, 93, 216, special issue on “Program Generation, Optimization, and Platform Adaptation”
 Fukugita, M., Nakamura, O., Okamura, S., Yasuda, N., Barentine, J. C., Brinkmann, J., Gunn, J. E., Harvanek, M., Ichikawa, T., Lupton, R. H., Schneider, D. P., Strauss, M. A., & York, D. G. 2007, *AJ*, 134, 579
 Gadotti, D. A. 2008, *MNRAS*, 384, 420
 Graham, A. W., Erwin, P., Trujillo, I., & Asensio Ramos, A. 2003, *AJ*, 125, 2951
 Gutiérrez, L., Erwin, P., Aladro, R., & Beckman, J. E. 2011, *AJ*, 142, 145
 Humphrey, P. J., Liu, W., & Buote, D. A. 2009, *ApJ*, 693, 822
 Kümmel, M., Koppenhoefer, J., Riffeser, A., Mohr, J., Desai, S., Henderson, R., Paech, K., & Wetzstein, M. 2013, in *Astronomical Society of the Pacific Conference Series*, Vol. 475, *Astronomical Data Analysis Software and Systems XXII*, ed. D. N. Friedel, 357
 Levenberg, K. 1944, *Quarterly J. Applied Mathematics*, 2, 164
 Liddle, A. R. 2007, *MNRAS*, 377, L74
 Llacer, J. & Nuñez, J. 1991, in *The Restoration of HST Images and Spectra*, ed. R. L. White & R. J. Allen, 62
 MacArthur, L. A., Courteau, S., & Holtzman, J. A. 2003, *ApJ*, 582, 689
 Markwardt, C. B. 2009, in *Astronomical Society of the Pacific Conference Series*, Vol. 411, *Astronomical Data Analysis Software and Systems XVIII*, ed. D. A. Bohlender, D. Durand, & P. Dowler, 251
 Marquardt, D. 1963, *J. SIAM*, 11, 431
 Moffat, A. F. J. 1969, *A&A*, 3, 455
 Monnet, G., Bacon, R., & Emsellem, E. 1992, *A&A*, 253, 366
 Moré, J. J. 1978, in *Lecture Notes in Mathematics*, Vol. 630, *Numerical Analysis*, ed. G. Watson (Springer Berlin Heidelberg), 105–116
 Muñoz-Mateos, J. C., Sheth, K., Gil de Paz, A., Meidt, S., Athanassoula, E., Bosma, A., Comerón, S., Elmegreen, D. M., & Elmegreen, B. G. 2013, *ApJ*, 771, 59
 Nelder, J. A. & Mead, R. 1965, *The Computer Journal*, 7, 308
 Nousek, J. A. & Shue, D. R. 1989, *ApJ*, 342, 1207
 Nuñez, J. & Llacer, J. 1993, *PASP*, 105, 1192
 Peng, C. Y., Ho, L. C., Impey, C. D., & Rix, H.-W. 2002, *AJ*, 124, 266
 —. 2010, *AJ*, 139, 2097
 Pohlen, M., Balcells, M., Lütticke, R., & Dettmar, R.-J. 2004, *A&A*, 422, 465
 Pohlen, M., Dettmar, R., Lütticke, R., & Schwarzkopf, U. 2000, *A&AS*, 144, 405

- Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 1992, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Ed. (Cambridge: Cambridge U. Press)
- Pritchett, C. & Kline, M. I. 1981, *AJ*, 86, 1859
- Richings, A. J., Uttley, P., & K rding, E. 2011, *MNRAS*, 415, 2158
- Rusli, S. P., Erwin, P., Saglia, R. P., Thomas, J., Fabricius, M., Bender, R., & Nowak, N. 2013, *MNRAS*, submitted
- Saglia, R. P., Bertschinger, E., Baggeley, G., Burstein, D., Colless, M., Davies, R. L., McMahan, Jr., R. K., & Wegner, G. 1993, *MNRAS*, 264, 961
- Schwarz, G. 1978, *Ann. Statist.*, 6, 461
- Scorza, C. & Bender, R. 1990, *A&A*, 235, 49
- S rsic, J. L. 1968, *Atlas de galaxies australes*
- Shaw, M. A. & Gilmore, G. 1989, *MNRAS*, 237, 903
- Sheth, K., Regan, M., Hinz, J. L., Gil de Paz, A., Men ndez-Delmestre, K., Mu oz-Mateos, J.-C., Seibert, M., Kim, T., Laurikainen, E., Salo, H., & et al. 2010, *PASP*, 122, 1397
- Simard, L. 1998, in *Astronomical Society of the Pacific Conference Series*, Vol. 145, *Astronomical Data Analysis Software and Systems VII*, ed. R. Albrecht, R. N. Hook, & H. A. Bushouse, 108
- Simard, L., Willmer, C. N. A., Vogt, N. P., Sarajedini, V. L., Phillips, A. C., Weiner, B. J., Koo, D. C., Im, M., Illingworth, G. D., & Faber, S. M. 2002, *ApJS*, 142, 1
- Spitzer, Jr., L. 1942, *ApJ*, 95, 329
- Storn, R. & Price, K. 1997, *Journal of Global Optimization*, 11, 314
- Takeuchi, T. T. 2000, *Ap&SS*, 271, 213
- Trujillo, I., Aguerri, J. A. L., Cepa, J., & Guti rrez, C. M. 2001, *MNRAS*, 328, 977
- Trujillo, I., Erwin, P., Asensio Ramos, A., & Graham, A. W. 2004, *AJ*, 127, 1917
- Tully, R. B., Rizzi, L., Shaya, E. J., Courtois, H. M., Makarov, D. I., & Jacobs, B. A. 2009, *AJ*, 138, 323
- van der Kruit, P. C. 1988, *A&A*, 192, 117
- van der Kruit, P. C. & Searle, L. 1981, *A&A*, 95, 105
- Xilouris, E. M., Alton, P. B., Davies, J. I., Kylafis, N. D., Papamastorakis, J., & Trewhella, M. 1998, *A&A*, 331, 894
- Xilouris, E. M., Byun, Y. I., Kylafis, N. D., Paleologou, E. V., & Papamastorakis, J. 1999, *A&A*, 344, 868
- Xilouris, E. M., Kylafis, N. D., Papamastorakis, J., Paleologou, E. V., & Haerendel, G. 1997, *A&A*, 325, 135
- Yoachim, P. & Dalcanton, J. J. 2006, *AJ*, 131, 226