

# DOLPHOT/MIRI User's Guide

version 2.0

Andrew Dolphin

adolphin@rtx.com

November 2022

<http://americano.dolphotsim.com/dolphot>

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Installation</b>	<b>3</b>
2.1	Makefile . . . . .	3
2.2	PSFs . . . . .	3
<b>3</b>	<b>Preprocessing Steps</b>	<b>4</b>
3.1	Create/Select Reference Images . . . . .	4
3.2	<i>mirimask</i> : Convert to DOLPHOT Usable Format . . . . .	4
3.3	<i>imarith</i> : Add Constant Background to Image . . . . .	5
3.4	<i>calcsky</i> : Calculate Sky for Image . . . . .	5
<b>4</b>	<b>Running DOLPHOT</b>	<b>7</b>
4.1	DOLPHOT Parameters . . . . .	7
4.2	DOLPHOT Output . . . . .	9
<b>5</b>	<b>PSF Utility Reference</b>	<b>10</b>
5.1	<i>mirimakepsf</i> : Compute PSF for Filter . . . . .	10
5.2	<i>mirishowpsf</i> : Display PSF for Filter . . . . .	10
<b>6</b>	<b>Recipe for MIRI Photometry</b>	<b>11</b>
<b>7</b>	<b>Changes to DOLPHOT/MIRI</b>	<b>12</b>

# 1 Introduction

This manual describes installation and use of the MIRI module for DOLPHOT. The MIRI module replaces the analytic PSF model with a lookup table computed using WebbPSF.

If you have any problems installing or using DOLPHOT or the provided utilities, please let me know.

Andrew Dolphin  
adolphin@rtx.com

## 2 Installation

Installation of the MIRI module is done in addition to the regular DOLPHOT installation. See the DOLPHOT manual for installation instructions for DOLPHOT.

The MIRI module requires less than 1Mb for the sources and binaries, plus less than 1Mb per set of PSFs. Note that the PSFs are distributed separately, as you only need PSFs for the filters that you will actually be using.

The .tar.gz file for the MIRI module (as well as the PSFs) should be expanded from the same directory from which the DOLPHOT distribution was expanded.

### 2.1 Makefile

The Makefile is the same as the regular DOLPHOT installation, but includes several options used by the MIRI module. To enable the MIRI module, the three lines beginning with the MIRI definition should be uncommented.

No customization of the Makefile within the “miri” subdirectory is needed.

After editing the makefile, you should be able to type “make” to compile DOLPHOT without any error or warning messages. Should you need to later recompile the programs, typing “make clean” will remove the original compilations to ensure that all programs get recompiled.

### 2.2 PSFs

You can either make your own PSFs or use the precomputed ones. Generally there is no need to generate your own, as precomputed PSFs are available for all filters. Instructions for computing PSFs are not currently available.

### 3 Preprocessing Steps

For the MIRI module to run correctly, data must be in the default format created by the STScI pipeline. Photometry is executed on the Stage 2 files (comparable to HST fit or flc images), while Stage 3 files (comparable to HST drz or drc) can be used as alignment reference images.

Because Stage 3 images produce suboptimal photometry (because of resampling of the images), support exists only for \_CFR and \_CAL datasets. DOLPHOT includes multiple-image support (including offset and rotation), so photometry can be run on multiple images of the same field, eliminating the primary purpose for photometering drizzled data.

This section takes you through the necessary steps (in order) for obtaining the best possible photometry from DOLPHOT.

#### 3.1 Create/Select Reference Images

If you have only one exposure per filter to photometer, and all images are taken at the same pointing, you can skip this step. Simply use your deepest image as the reference image in DOLPHOT.

It is possible that the MIRI reduction pipeline has already drizzled your images into a single drizzled image per filter. If this is the case, you can also forgo running drizzle, and use the deepest drizzled image as your reference frame.

Otherwise, there may be a need to generate a deep reference image in order to have sufficient depth and area coverage to align all images in your data set. Instructions on how to perform this are beyond the scope of this manual.

Once drizzling is complete, you may want to co-add the Stage 2 images taken at the same pointing using the DOLPHOT utility *imcombine*. (This can also improve cosmic ray rejection.) When you use *imcombine* for this purpose, make sure that the registration factor and sigma clipping are set sufficiently high that no additional cleaning is done, beyond the cleaning done by multidrizzle. It is also acceptable to photometer all of the Stage 2 images separately.

#### 3.2 *mirimask*: Convert to DOLPHOT Usable Format

```
mirimask <flags> <fits files>
```

*MIRImask* takes as input FITS files in the format created by the STScI pipeline. Stage 2 processed images consist of a single 2048x2048 chip, while Stage 3 processed images may include multiple chips in the same filter, resampled into a common coordinate frame. Both Stage 2 and Stage 3 data also contain multiple extensions providing data quality, pixel area, and other data processed by *MIRImask*.

It is important to note that many or most MIRI images have a roughly constant background level subtracted in the JWST pipeline, but that background level is not indicated in the resulting science project. This will result in DOLPHOT underestimating the photon noise in the image, as the subtracted background level is not evident in the

science image. This can be addressed in one of several ways. If the level of background to add is known, it can be simply added to the science image `imgadd` or a separate script, e.g. in python. A known background level can also be added *mirimask* by setting the noise floor to add (in units of DN) with the `-bg` flag set to a non-negative number. Finally, *mirimask* will estimate the value empirically and apply it if the `-bg` flag is set to a negative number.

The `-mask_lyot` flag will cause *mirimask* to mask out pixels in the Lyot coronagraph region.

The `-estnoise` flag will estimate the readout noise from the FITS readout noise extension, and use that value in place of the default readout noise (which is based on single-readout noise and the detector parameters). This is generally recommended, though should be compared to the default noise in the *mirimask* console output to verify the estimated noise is close to the default value. A significant deviation could be a sign of issues with one of the two.

The `-noetctime` flag will cause *mirimask* (and downstream programs including *dolphot*) to use the JWST EFFEXPTM in place of the default “start to end” calculation reported by the ETC. This is not recommended.

Before running *mirimask*, make sure you have backed up the original images, since *mirimask* will alter them!

A masked or saturated pixel is skipped by all other HSTphot routines - it is not used in sky determination, photometry, aperture corrections, etc.

All extensions other than SCI (the data extension) are deleted when writing the data back to disk. The output image will thus have only one extension.

### 3.3 *imarith*: Add Constant Background to Image

```
imarith <input fits file> + <bg value> <output fits file>
```

Most MIRI images have a sky image subtracted by the JWST processing pipeline, which lowers the reported counts in the SCI image, and thus the photon noise estimated by DOLPHOT. A straightforward way to remedy this is to add a constant value back to the image after processing by *mirimask* using the *imarith* utility. Note the output filename can be the same as the input filename, which will overwrite the original file with the background-added file.

### 3.4 *calcsky*: Calculate Sky for Image

```
calcsky <fits base> < $r_{in}$ > < $r_{out}$ > <step> < $\sigma_{low}$ > < $\sigma_{high}$ >
```

*Calcsky* is described in the DOLPHOT manual. Recommended parameters are as follows:

$$r_{in} = 10$$

$$r_{out} = 25$$

*step* = 2 for an accurate sky map, or *step* = -64 for a quick one.

$$\sigma_{low} = 2.25$$

$$\sigma_{high} = 2.00$$

Since I always use non-zero FitSky settings in my photometry, the negative values of *step* are sufficient.

Note that running *calcsky* is currently mandatory, even if using FitSky settings of 3 or 4.

## 4 Running DOLPHOT

### 4.1 DOLPHOT Parameters

The MIRI module of DOLPHOT will be invoked whenever processing images that have been run through mirimask. When using the MIRI module, several parameters are disabled or have restricted ranges.

img\_aprad is set to 10 pixels

img\_RPSF and img\_RAper cannot exceed 24 pixels

Zero disabled (set to VEGAMAG zero points)

FPSF, SubPixel, img\_psfa, img\_psfb, img\_psfsc disabled

EPSF, PSFsol, PSFStep disabled

MinS, MaxS, MaxE disabled

Recommended values for other parameters when running with `FitSky = 1` are as follows:

`PSFPhot = 1`

`PSFPhotIt = 2`

`Force1 = 0`

`img_apsky = 20 35`

`img_RAper = 5`

`img_RChi = 2.0`

`img_RSky = 15 35`

`img_RPSF = 15`

`SkipSky = 1`

`SkySig = 2.25`

`SecondPass = 5`

`SigFindMult = 0.85`

`MaxIT = 25`

`NoiseMult = 0.10`

`FSat = 0.999`



```

ApCor = 1
RCentroid = 1
PosStep = 0.25
RCombine = 1.5
SigPSF = 5.0
PSFres = 1
UseWCS = 2

```

Running with `FitSky = 2` is recommended in crowded fields. In this case, recommended parameter changes to the above are as follows:

```

img_RAper = 3
img_RSky2 = 4 10

```

If using `FitSky = 3`, the parameters are largely the same except that suggested `img_RAper` values are 8 pixels.

In some cases, blends or extended objects can hamper photometry of nearby stars. Setting `Force1 = 1` will solve this, but of course will also result in false detections of hot pixels and extended objects as single stars. So one may have to choose between a very clean but slightly incomplete CMD and a complete but contaminated one.

Finally, there are four new parameters that can be used:

```

FlagMask = 4
CombineChi = 0
InterpPSFlib = 1
MIRIvega = 1

```

`FlagMask` is a bitwise mask that determines what error flags will not be accepted when producing the combined photometry blocks for each filter. Note that error flag values of eight or more (when the “extreme case”) always cause the photometry to be ignored. A value of zero allows photometry with an error flag less than eight to be used. Adding one eliminates stars close to the chip edge, adding two eliminates stars with too many bad pixels, and adding four eliminates stars with saturated cores.

`CombineChi` also affects the combined photometry blocks. If set to zero (default), photometry will be combined weighted by  $1/\sigma^2$  to maximize signal to noise. If set to one, weights will be  $1/\sigma^2 \max(1, \chi^2)$  to reduce the impact of epochs with bad photometry. Note that using `CombineChi` of one will require tuning `NoiseMult` so that well measured stars have  $\chi = 1$  at all magnitudes (plots of chi vs. magnitude should show this). Note also that this will result in larger uncertainties for combined (but not individual

image) magnitudes and normalized count rates, as the individual image uncertainties are effectively multiplied by  $\chi$  when calculating combined magnitudes.

If `InterpPSFlib` is set to 0, the PSF library will use the nearest X,Y position where a precalculated PSF is available rather than interpolating. The impact is  $\approx 1\%$  on the PSF shape but some speed improvement.

If `MIRIvega` is set to 0, calibrated fluxes will be provided in units of Jy, and instrumental magnitudes in units of ABmag. Otherwise, when set to 1, Vega magnitudes are used and calibrated fluxes are scaled to for a zeroth magnitude source.

Note that the DOLPHOT/MIRI module does not have default distortion parameters included, but rather relies on the WCS information in the FITS file obtained from the JWST pipeline. Thus, `UseWCS = 2` is required when reducing MIRI data.

## 4.2 DOLPHOT Output

The DOLPHOT/MIRI output is virtually identical to the regular DOLPHOT output, with a few exceptions. First, if multiple images exist per filter, it will insert extra sets of photometry for each such filter (in order of wavelength) prior to the individual-image photometry. Following any combined-filter photometry are the single-image photometry blocks, in order that the images are provided in the parameter files.

Within the photometry blocks, there are also some differences. First is that calibrated instrumental magnitudes and transformed magnitudes are provided. Note that transformations are not yet supported, so the second magnitude is always 99.999 in the current DOLPHOT release. Also, a calibrated count rate and uncertainty is provided (effectively the instrumental magnitude and uncertainty expressed as a count rate).

As with all photometry, you will need to trim your detection list to eliminate objects classified as non-stellar, low signal-to-noise, or bad photometry quality. See the DOLPHOT manual for further discussion.

## 5 PSF Utility Reference

The following programs are utilities included with the MIRI module. Since there is little need for computing custom PSFs, no documentation is provided – use at your own risk!

### 5.1 *mirimakepsf*: Compute PSF for Filter

```
mirimakepsf <filter>
```

### 5.2 *mirishowpsf*: Display PSF for Filter

```
mirishowpsf <filter> <X/512> <Y/512> <dX> <dY> <<flags>>
```

## 6 Recipe for MIRI Photometry

Assuming that an appropriate reference image exists, the recipe for generating photometry is reasonably simple.

1. Run mirimask on all MIRI images
2. Run calsky on each image to generate \*.sky.fits files
3. Run dolphot

## 7 Changes to DOLPHOT/MIRI

December 2, 2022 release

1. Initial release
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.
3. Caveat: The cruciform structure has not yet been incorporated into the optical model used to generate the PSF models
4. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
5. Not yet tested: comparison of observational to synthetic CMDs, and PSF time variability

December 3, 2022 release

1. Added ability to select between flux and magnitude outputs in Vega mag vs. Jy using the MIRIvega parameter.
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.
3. Caveat: The cruciform structure has not yet been incorporated into the optical model used to generate the PSF models
4. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
5. Not yet tested: comparison of observational to synthetic CMDs, and PSF time variability

April 6, 2023 release

1. Corrected error in PSF library that impacted PSF variability across chips. Note users will need to re-download PSFs.
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.

3. Caveat: The cruciform structure has not yet been incorporated into the optical model used to generate the PSF models
4. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
5. Not yet tested: comparison of observational to synthetic CMDs, and PSF time variability

December 2, 2023 release

1. -etctime flag added to mirimask to set DOLPHOT exposure time consistently with JWST ETC.
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.
3. Caveat: The cruciform structure has not yet been incorporated into the optical model used to generate the PSF models
4. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
5. Not yet tested: comparison of observational to synthetic CMDs, and PSF time variability

December 2, 2023 release

1. Updated “mirimask” utility based on pipeline updated data quality flags for coronagraph regions.
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 (general masked) or 512 (coronagraph) is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.
3. Caveat: The cruciform structure has not yet been incorporated into the optical model used to generate the PSF models
4. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
5. Not yet tested: comparison of observational to synthetic CMDs, and PSF time variability

February 4, 2024 release

1. Updated all PSF libraries based on the current WebbPSF (1.2.1), which matches empirical PSFs significantly better.
2. Caveat: Data Quality processing not fully implemented for Stage 2 images; only saturated pixels are used, while a flag bit of 262144 (general masked) or 512 (coronagraph) is used as an indicator of a masked-out pixel. Otherwise, stage 2 pixels with SCI values of exactly zero are masked as bad pixels, and all other pixels are assumed good.
3. Caveat: All alignment and calibration data are obtained from the FITS headers; the ASDF data are not used.
4. Not yet tested: impact from PSF time variability

January 13, 2025 release

1. Changed default exposure time to use ETC calculation
2. Improved default readout noise calculation
3. Added empirical readout noise estimator to mirimask
4. Added ability to account for Poisson noise pedestal from background frame subtraction, either by estimation or by user-provided value, into mirimask