Cookbook for Measuring Photometry and Deriving Galaxy Quantities for WALLABY Sources

Version 3.0 – 2022-11-24

T. Reynolds

Contents

1 Introduction

This document describes the methods and scripts used to measure galaxy properties for H I sources detected by SoFiA using H I source data products and ancillary multi-wavelength (e.g. ultraviolet, optical and infrared) image cutouts. I describe how to measure quantities from the H I data in Section 2. Sections 3 and 4 describe how to download multi-wavelength image data and measure photometry in the multi-wavelength band images, respectively. I describe how to derive stellar masses and star formation rates using the measured photometry in Section 5. In Section 6, I provide a step-by-step guide of the order in which to run the scripts and their respective settings to measure and derive galaxy quantities from all imaging bands. Table 1 lists the PYTHON scripts used for the different analysis steps described in this document.

File Name	Description
`create_directories.py`	Creates all directory paths
`download_panstarrs.py`	Download PanSTARRS images
`download_galex.py`	Download *GALEX* images
`download_wise.py`	Download unWISE images
`photometry_wallaby.py`	Measure HI radial profiles and HI quantities
`photometry_panstarrs.py`	Measure optical photometry and quantities
`photometry_galex.py`	Measure UV photometry and quantities
`photometry_wise.py`	Measure IR photometry and quantities
`derived_properties.py`	Derive physical quantities from photometry
`galaxy_summary_plots.py`	Create summary plots

Table 1. List of PYTHON scripts.

1.1 Source Classification

Not all H I detections and optical images are of sufficient quality to use for scientific analysis. I have created flags to classify various issues with either the H I detections or optical PanSTARRS images, which are listed in Tables 2 and 3, respectively. These flags are included in tables titled `*_catalogue_flags.fits`, where `*` is the team release name.

Flag	Description	Example
0	No optical counterpart in r-band image	Figure 1(a)
1	Single optical counterpart	Figure 1(b)
2	Multiple optical counterparts (i.e. interacting system)	Figure 1(c)
3	Shredded H I source (i.e. only half of galaxy detected)	Figure 2(a)/2(b)
4	Combined shredded H I source	—
5	Artefacts present in H I detection (i.e. continuum residual)	Figure 1(d)

Table 2. Flags for classifying the H I data: `flag_src_class`.

Flag	Description	Example
0	Successfully found optical counterpart for photometry	Figure 1(b)
1	Segmentation map to be fix (change to 0 once done)	—
3	Foreground star/s coincident with optical counterpart	Figure 3(a)
5	Artefacts present in r-band image	Figure 3(b)/3(c)

Table 3. Flags for classifying the PanSTARRS r-band image data: `flag_opt_fit`.

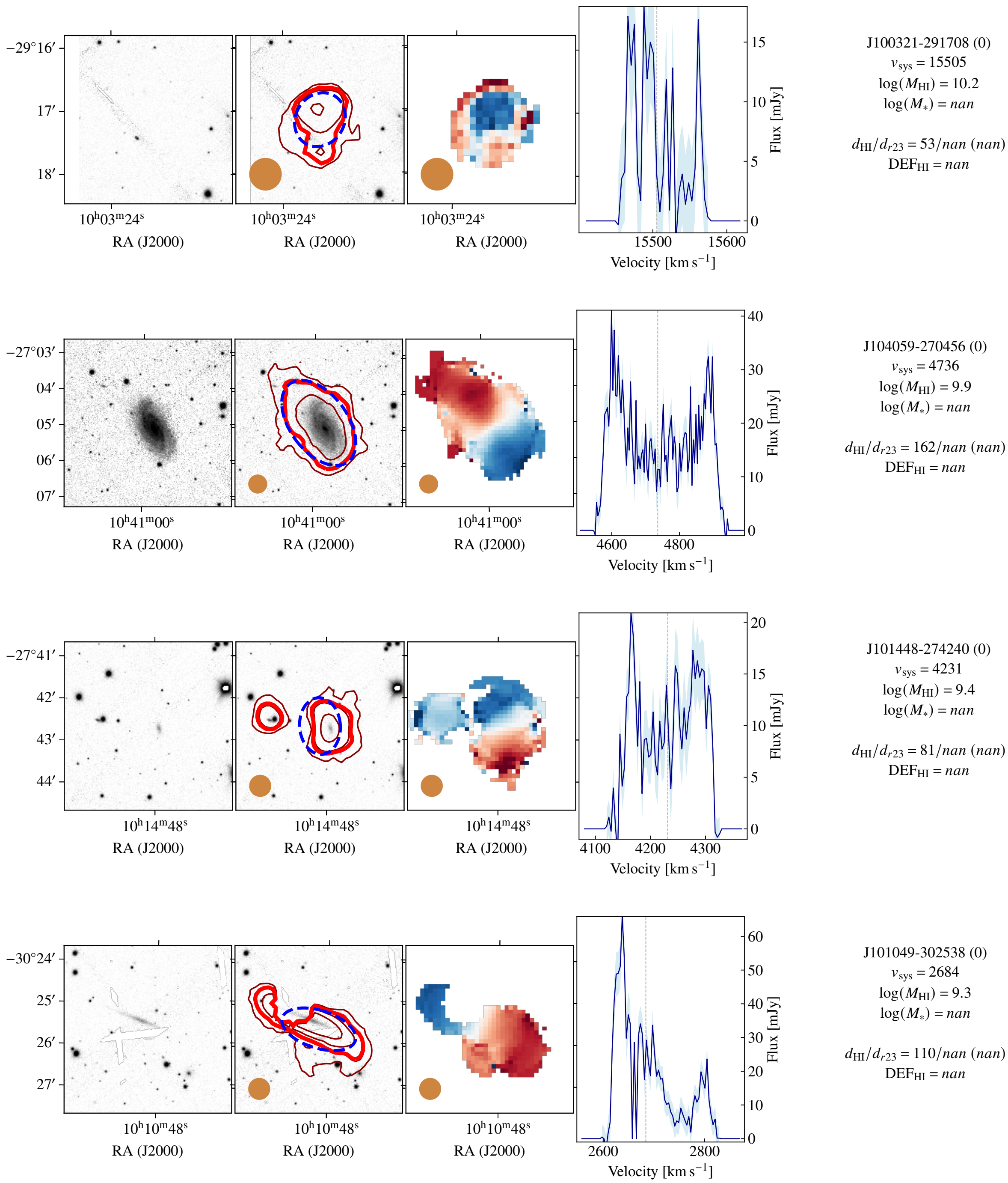

Figure 1. Summary plots showing r-band image, r-band image with overlaid H I contours, moment 1 map and integrated H I profile for example H I flag classifications. Top row: example of no optical counterpart. Second row: single optical counterpart. Third row: multiple optical counterparts. Fourth row: artefacts in H I detection (masked continuum residual).

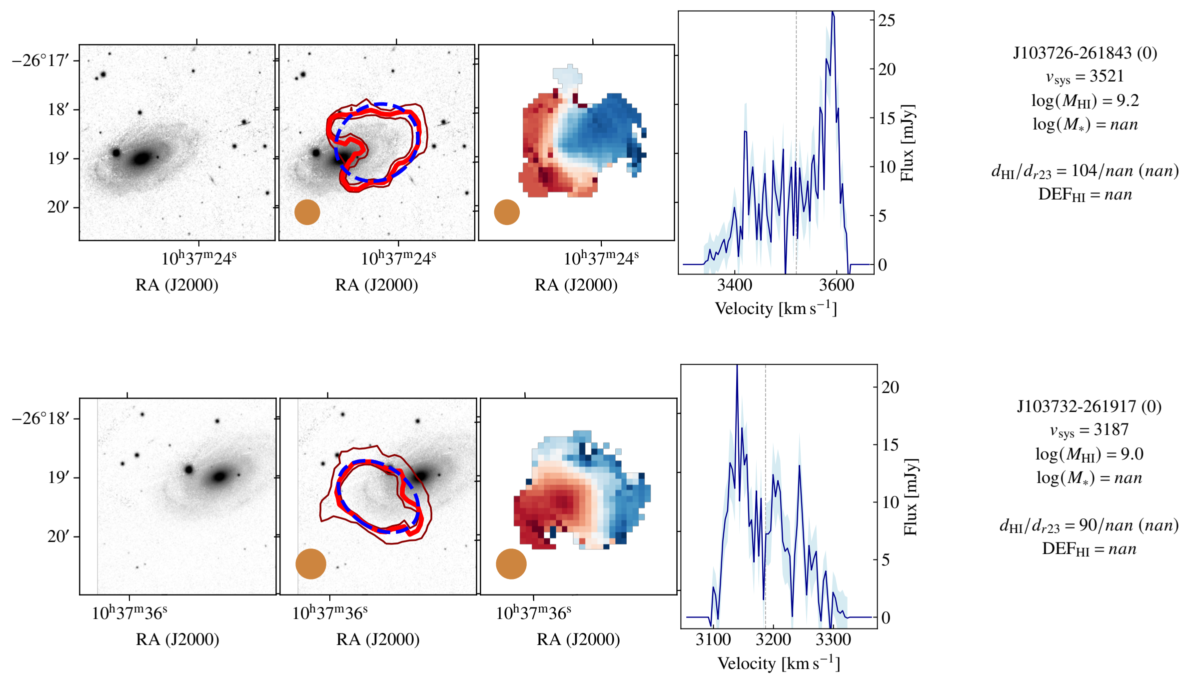

Figure 2. Similar to Figure 1(a) showing the two halves of a shredded H I source.

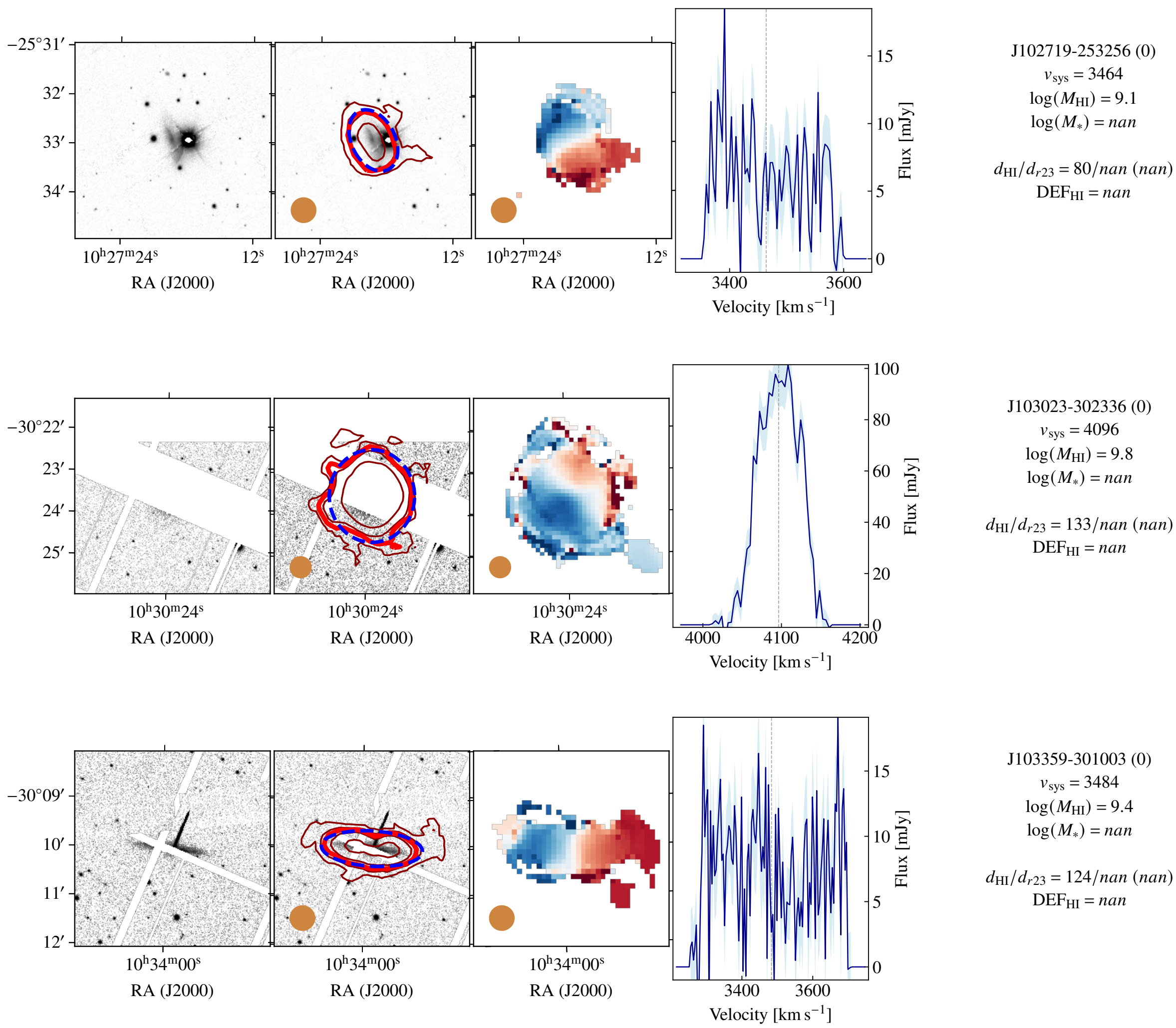

Figure 3. Similar to Figure 1(a) showing examples of optical flag classifications. Top row: foreground star impacting optical counterpart. Third/fourth rows: artefacts in PanSTARRS image.

2 HI Data

2.1 Data Products

The H I data products are produced from the SoFiA source finding runs by Tobias Westmeier and Austin Shen and are available from the WALLABY database[1]. SoFiA produces a source catalogue with basic source properties (e.g. RA, Dec, systemic frequency, integrated flux, etc.) and associated data products (e.g. integrated spectrum, moment 0, 1 and 2 maps, spectral line cubelet, etc.). Westmeier et al. (2022) provides descriptions of the source catalogue and associated data products. Meyer et al. (2017) provide a comprehensive list of equations and conversions for calculating physical quantities (e.g. H I mass and luminosity distance) from the SoFiA source properties.

2.2 HI Properties

The integrated intensity (moment 0) map is used to measure the H I radial surface density profile, from which the size of the H I disc can be measured. I have written a PYTHON script for extracting the radial profile by measuring the average H I surface density in annuli overlaid on the moment 0 map, which I describe below.

2.2.1 WALLABY Flux Correction

The majority of WALLABY sources are detected due to spatially and spectrally smoothing the spectral line cube at various scales with SoFiA to increase the cube signal to noise ratio (SNR). As a result, many sources may not be deconvolved as they are too faint to be detected in individual channels during the deconvolution step of the ASKAPsoft processing pipeline (for further details see Westmeier et al. 2022). This causes the majority of integrated H I flux measurements to be underestimated, with fainter H I detections being more severely affected. I apply the statistical correction to the integrated fluxes proposed in Westmeier et al. (2022) presented in their equations 9 and 10.

$$\log(S_{\mathrm{corr}}) = \log(S) - 0.0285\log(S)^3 + 0.439\log(S)^2 - 2.294\log(S) + 4.097, \tag{1}$$

where S and S_{corr} are the measured and corrected HI fluxes in Jy Hz. The correction is $< 10\%$ for fluxes $> 10^5$ Jy Hz, but increases from $\sim 25\%$ to $\sim 60\%$ as the measured flux decreases from $\sim 10^{4.5}$ to $\sim 10^3$ Jy Hz. I also apply this correction to the radial surface brightness profiles measured from the moment 0 maps by scaling the profiles by the ratio between the measured and corrected integrated fluxes.

2.2.2 HI Radial Profiles

Determining the size of the H I disc requires measuring the H I radial surface density profile from which both isodensity and effective radii can be measured. To do this, I measure the total flux in elliptical annuli from the moment 0 map.

The annuli to be overlaid on the moment 0 map need to have a defined centre, position angle and inclination angle (or major and minor axes lengths). I obtain these parameters by fitting

[1] Instructions on accessing the WALLABY database are available on the WALLABY internal wiki here: https://pm.atnf.csiro.au/askap/projects/sup-wallaby/wiki/WALLABY_Data_Access.

a 2-dimensional Gaussian to the moment 0 map (using the assumption that the moment 0 map can be modelled as a 2-dimensional Gaussian). I use the ASTROPY package in PYTHON to fit a model 2-dimensional Gaussian created with the GAUSSIAN2D task using the LEVMARLSQFITTER task, which using the Levenberg-Marquardt algorithm, to the moment 0 map. The 2-dimensional Gaussian fit returns the x, y position centre, the length of the major and minor axes and the position angle. I fit a 2-dimensional Gaussian to the H I moment 0 map rather than a fit to an optical image as this allows for all H I sources to be measured in a consistent manner and does not require an optical counterparts.

I next use the PYTHON package PHOTUTILS to measure the surface density in annuli on the moment 0 map. I use the PHOTUTILS functions ELLIPTICALAPERTURE and ELLIPTICALANNULUS to define sets of elliptical apertures and annuli, respectively, separated by 7.5 arcsec for which I extract the total flux density using the task APERTURE_PHOTOMETRY. I also provide APERTURE_PHOTOMETRY with the local root mean square (RMS) noise from the SoFiA source catalogue, which then returns an uncertainty in the measured flux density in each aperture/annulus. The distinction between an aperture and an annulus is that an aperture measures the total flux enclosed within the ellipse (i.e. from the centre) while an annulus measures the total flux within a ring of width 7.5 arcsec.

I convert the extracted flux densities (Jy Hz pixel^{-2}) to surface densities (M$_\odot$ pc^{-2}) using equations 76 and 81 from Meyer et al. (2017):

$$\frac{\Sigma_{\rm HI}}{\rm M_\odot\,pc^{-2}} = (8.01 \times 10^{-21})(2.33 \times 10^{20})(1+z)^4 \left(\frac{S}{\rm Jy\,Hz}\right)\left(\frac{ab}{\rm arcsec}\right)^{-1}, \quad (2)$$

where S is the flux density and a, b are the synthesised beam major and minor axes ($a, b \sim 30$ arcsec for WALLABY; the exact major and minor axis values are taken from the FITS file header). These surface densities are the total surface densities in each aperture/annulus. I compute the average surface density in each aperture/annulus by dividing the total by the area of the corresponding aperture/annulus.

I correct the surface density profile for inclination using $\cos(i)$, where i is the inclination angle determined from the 2-dimensional Gaussian fit and the emission is assumed to be optically thin. I note that for galaxies with high inclinations and/or that are marginally resolved, the WALLABY synthesised beam will dominate the minor axis size. I correct for this effect by deconvolving the fitted Gaussian by the WALLABY synthesised beam before determining the inclination angle. Assuming the galaxy's H I distribution at this surface density is symmetric, the H I disc diameter is $d_{\rm HI,0} = 2r_{\rm HI}$, where $d_{\rm HI,0}$ has not been corrected for the effect of the synthesised beam size. Assuming the beam and H I disc can both be approximated as Gaussians, this correction for the synthesised beam takes the form,

$$d_{\rm HI} = \sqrt{d^2_{\rm HI,0} - ab}. \quad (3)$$

2.2.3 HI within Optical Disc

After measuring PanSTARRS photometry and surface brightness profiles (Section 4.1), from which magnitudes and optical sizes are measured, the H I within the optical disc can be measured. I measure the total H I flux and average H I surface density within optical apertures defined by the optical radius and the optical segment position angle and axis ratio using the PHOTUTILS package task ELLIPTICALAPERTURE. I measure the inner H I flux and surface density for apertures defined by the isophotal 25 mag arcsec^{-2} radius, the effective (r_{50}) radius and the r_{90} radius. These will not be accurate for sources with optical apertures smaller than the ASKAP synthesised beam (30 arcsec) along either the major or minor axes (e.g. some highly inclined galaxies may have a major axis of > 30 arcsec, but a minor axis of < 30 arcsec) and selection cuts are required prior to scientific analysis.

3 Multi-wavelength Image Download

I have written separate PYTHON scripts for creating and downloading PanSTARRS (optical), *GALEX* (ultraviolet) and WISE (infrared) image cutouts. I have modified a PYTHON script provided by the PanSTARRS survey for creating image cutouts[2]. For *GALEX* and WISE, the scripts use functions written by Li Shao to access *GALEX* and WISE online imaging data archives. Li Shao's functions require SWARP[3] (Bertin et al. 2002), which resamples and coadds individual image frames to create the final image cutouts in the *GALEX* and WISE image bands. I have installed and been using SWARP version 2.41.5. The required input information for these scripts are the H I source position (RA and Dec), which is taken from the SoFiA source catalogue.

The available photometric bands for each wavelength are:

- PanSTARRS: g-, r-, i-, y-, z-bands (PSF: 1.5 arcsec)
- *GALEX*: NUV-, FUV-bands (PSF: 4.9, 4.2 arcsec, respectively)
- WISE: W1-, W2-, W3-, W4-bands (PSF: [6.08, 5.60], [6.84, 6.12], [7.36, 6.08], [11.99, 11.65] arcsec, respectively [major axis, minor axis])

3.1 PanSTARRS

PanSTARRS images are created using a script that creates an image cutout URL, which is then downloaded using WGET. The function `geturl()` takes as input the position RA and Dec, the size of the image in pixels, the PanSTARRS image band and the file format (i.e. FITS). I define the size of the cutout image by the larger of the SoFiA moment 0 FITS file `'NAXIS1'` and `'NAXIS2'`. This requires having access to the SoFiA moment 0 FITS file. If these are not available then I have provided the option to use a fixed image size of 512 pixels, however this will be too small for some objects, which will require a larger image size. I convert to the number of PanSTARRS pixels by

$$s_{\rm opt} = s_{\rm HI} \times a_{\rm pixel} \times 3600/0.25, \tag{4}$$

where $s_{\rm opt}$ is the PanSTARRS image size in pixels, $s_{\rm HI}$ is the larger of `'NAXIS1'` and `'NAXIS2'`, $a_{\rm pixel}$ is the moment 0 map pixel scale (degrees/pixel), 3600 converts from degrees to arcseconds and 0.25 arcsec/pixel is the approximate PanSTARRS pixel scale (exact scale is 0.2498 arcsec/pixel). I use a minimum image size of 240 pixels (i.e. if $s_{\rm opt} < 240$ pixels then I set $s_{\rm opt} = 240$ pixels). This is to ensure there is sufficient sky included in the image for measuring the local sky background.

Once downloaded, the PanSTARRS FITS image headers must be edited as they contain header keywords which ASTROPY ignores, but are required to correctly determine the FITS world coordinate system (WCS) information. The following keywords should be deleted:

[2] https://ps1images.stsci.edu/ps1image.html

[3] https://www.astromatic.net/software/swarp/. SWARP is installed in '/usr/local/bin' by default. If this directory is included in your path then the python scripts will be able to find and run SWARP without needing to specify the location of SWARP specifically.

- 'PC001001', 'PC001002', 'PC002001', 'PC002002'

and the sign of the keyword 'CDELT1' must be switched (i.e. if 'CDELT1' is negative then it must be made positive: `hdr['CDELT1'] = -1 * hdr['CDELT1']`). The PanSTARRS FITS files use the 'PC*' keywords in conjunction with the 'CDELT*' keywords to correctly specify the WCS, which works in DS9, but not in PYTHON or KVIS.

3.2 GALEX

The *GALEX* download script relies on a CSV table that contains the RA and Dec position centre and the full path directory of each observed *GALEX* tile (`galex_images.csv.gz`, located in the same directory as the PYTHON scripts) within the online data store[4] from *GALEX* DR6+7 (Bianchi et al. 2017). The first step of the script downloads all *GALEX* tiles covering the chosen cutout region into a temporary directory. The function `get_galex_image()` takes as input the position RA and Dec, the size of the image in degrees (I use a fixed size of 0.15 degrees), base output FITS file names to which '-nuv.fits' or '-fuv.fits' will be appended, whether to keep to delete the individual full downloaded *GALEX* tiles and the name of the directory into which the full tiles will be downloaded. For each tile, a weight map is created and the sky background is derived and subtracted. Then the specified region is cutout and mosaicked using SWARP.

GALEX does not have complete sky coverage hence there will not be *GALEX* imaging for some WALLABY detections. The majority of galaxies only have shallow *GALEX* All-sky Imaging Survey (AIS) data, with smaller numbers being included in the deeper Medium Imaging Survey (MIS) and Nearby Galaxy Survey (NGS).

3.3 WISE

The process for WISE images is similar to that for *GALEX*. The WISE download script relies on a FITS table that contains the RA and Dec position centre and the full path directory of each observed WISE tile (`unwise_tiles.fits.gz`, located in the same directory as the PYTHON scripts) within the online data store[5]. I use the unWISE co-added images. Unlike the original WISE images, the unWISE co-adds retain the intrinsic resolution of the data (Lang 2014; Meisner et al. 2017). The first step of the script downloads all WISE tiles covering the chosen cutout region into a temporary directory. The function `get_unwise_image()` takes as input the position RA and Dec, the size of the image in degrees (I use a fixed size of 0.25 degrees), base output FITS file names to which '-w1.fits', '-w2.fits', '-w3.fits' or '-w4.fits' will be appended, whether to keep to delete the individual full downloaded WISE tiles and the name of the directory into which the full tiles will be downloaded. The specified region is then cutout and mosaicked using SWARP.

[4] http://galex.stsci.edu/data

[5] W1 (http://unwise.me/data/neo6/unwise-coadds/fulldepth) and W3 and W4 (http://unwise.me/data/allwise/unwise-coadds/fulldepth).

4 Measuring Photometry

4.1 PanSTARRS

I measure g- and r-band photometry using the PYTHON package PHOTUTILS. The majority of the image analysis is carried out on the r-band image and is then applied to the g-band image. The basic steps for measuring the r-band photometry are as follows:

1. Create a segmentation map and mask non-target objects in the image (top row, Figure 4(a)).
2. Use the target object segment to define the position centre (x, y), position angle (θ) and axis ratio (b/a) of annuli and apertures for measuring flux (middle row, Figure 4(b)).
3. Measure and subtract the local sky background in annuli around the target galaxy (middle row, Figure 4(b)).
4. Measure aperture and annulus photometry from background subtracted image (middle row, Figure 4(b)).
5. Measure total optical magnitudes and optical sizes from aperture and annulus photometry (bottom row, Figure 4(c)).

I then use r-band image segmentation mask and defined annuli/apertures to carry out steps 3–5 for the g-band image.

4.1.1 Image Segmentation

The first step is to identify and mask all objects in the r-band image other than the target galaxy. To do this, I create a segmentation map using the task SEGMENTATION. I first mask all pixels with values $> 1\,000$ before measuring and subtracting the average background across the image ($\sigma_{\rm bkgd}$). I use this background-subtracted image for creating the segmentation map. In the original image, I measure the average pixel value, $\bar{x}$, in a 10-by-10 pixel box centred on the centre position of the 2d Gaussian fit to the H I moment 0 map. I then create a segmentation map which identifies sources with emission greater than a given multiple of the image background, $\sigma_{\rm bkgd}$:

- If $\bar{x} > 5\sigma_{\rm bkgd}$, threshold $= 3.5\sigma_{\rm bkgd}$
- If $\bar{x} > 2\sigma_{\rm bkgd}$, threshold $= 2\sigma_{\rm bkgd}$
- If $\bar{x} < 2\sigma_{\rm bkgd}$, threshold $= 0.75\sigma_{\rm bkgd}$

where the threshold is determined by the $\bar{x}$ value. The purpose of having multiple thresholds is to improve the success rate of separating nearby objects into individual segments and identifying faint sources. To identify the segment corresponding to the target galaxy, I identify the segment with the smallest separation to the H I map Gaussian fit that has a radius > 10 pixels. If this segment has a separation > 30 arcsec or there are no segments with radii > 10 pixels, then I identify the segment with the smallest separation that has a radius > 5 pixels. I then mask all other segments.

This is successful at identifying the target galaxy in $\sim 89\%$ of cases. However, the rest require some manual intervention due to sources being confused, an incorrect object being identified by the selection criteria (separation/minimum size) or the target source being fainter than the threshold

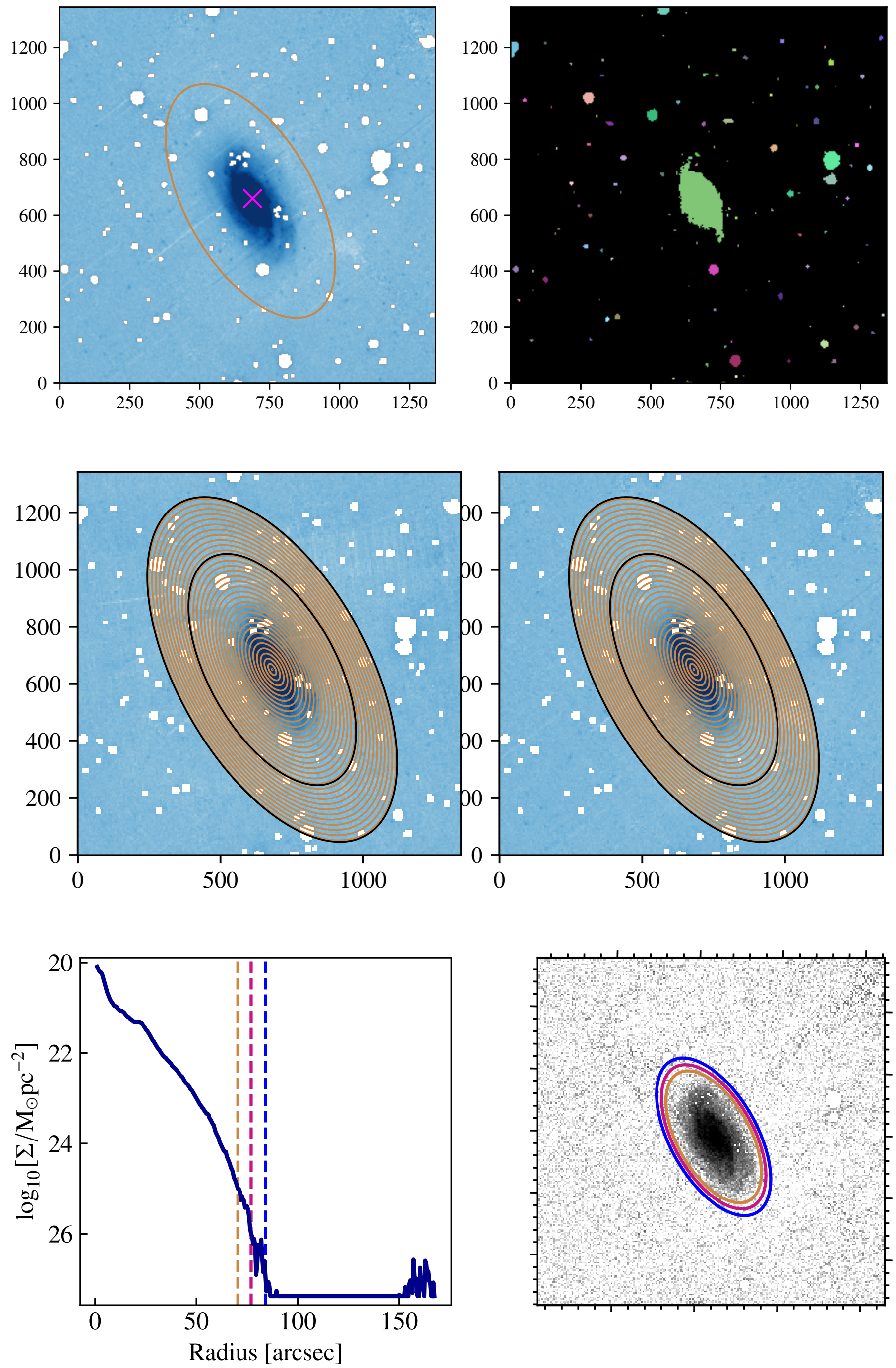

Figure 4. The steps of measuring optical photometry and radii. Top row: the masked r-band image (left) and the segmentation map (right). The pink cross indicates the centre of 2D Gaussian fit to the H I moment 0 map and the orange ellipse is defined by the target segment with the segment radius increased by a factor of 10. Middle row: ellipses overlaid on g- and r-band images indicating every 10^{th} annulus/aperture (left and right, respectively) with the black ellipses indicating the annuli range used to measure the local sky background. Bottom row: the radial r-band surface brightness profile with dashed lines at the radii where the surface brightness reaches 25, 26 and 27 mag arcsec^{-2} (left) and the r-band image with overlaid ellipses showing these radii (right).

limits. The majority of the failed cases are due to nearby objects being included in the target galaxy's segment. Segments in the map can be deblended into multiple sources using the task DEBLEND_SOURCES (e.g. Figure 5). The DEBLEND_SOURCES task is not always successful and a better solution is to set a higher threshold (e.g. $5\sigma_{\mathrm{bkgd}}$) during the image segmentation process, which will then identify the confused sources as separate segments (e.g. Figure 6). Wrong identification based on separation/size can generally be fixed by decreasing or increasing the minimum size (e.g. > 2 or > 30 pixels, e.g. Figures 7 and 8, respectively). If the target source is missed during the image segmentation, the solution is to set a lower threshold (e.g. $0.5\sigma_{\mathrm{bkgd}}$, e.g. Figure 9).

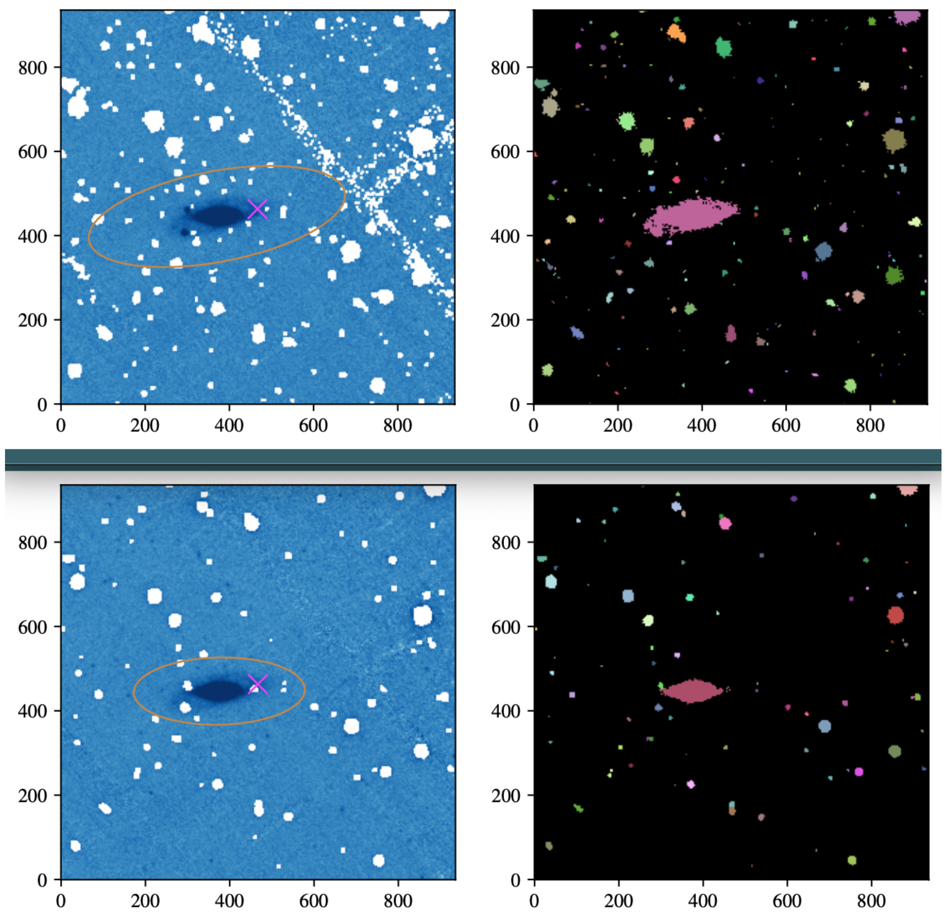

Figure 5. Top row: An example (J102207-282201) where the initial segmentation map does not find the target segment. Bottom row: Result from deblending using the DEBLEND_SOURCES task. Panels are as in Figure 4(a), top row.

4.1.2 Image Photometry

The masked r-band image is used for measuring the r-band photometry and this mask is also applied to the g-band image prior to measuring the g-band photometry. I fit a series of elliptical apertures and annuli to the r- and g-band images using the position centre, position angle and axis ratio of the target segment. The apertures and annuli are defined using the PHOTUTILS package tasks EllipticalAperture and EllipticalAnnulus. The total emission within each aperture/annulus is then found using the task APERTURE_PHOTOMETRY. The first step when

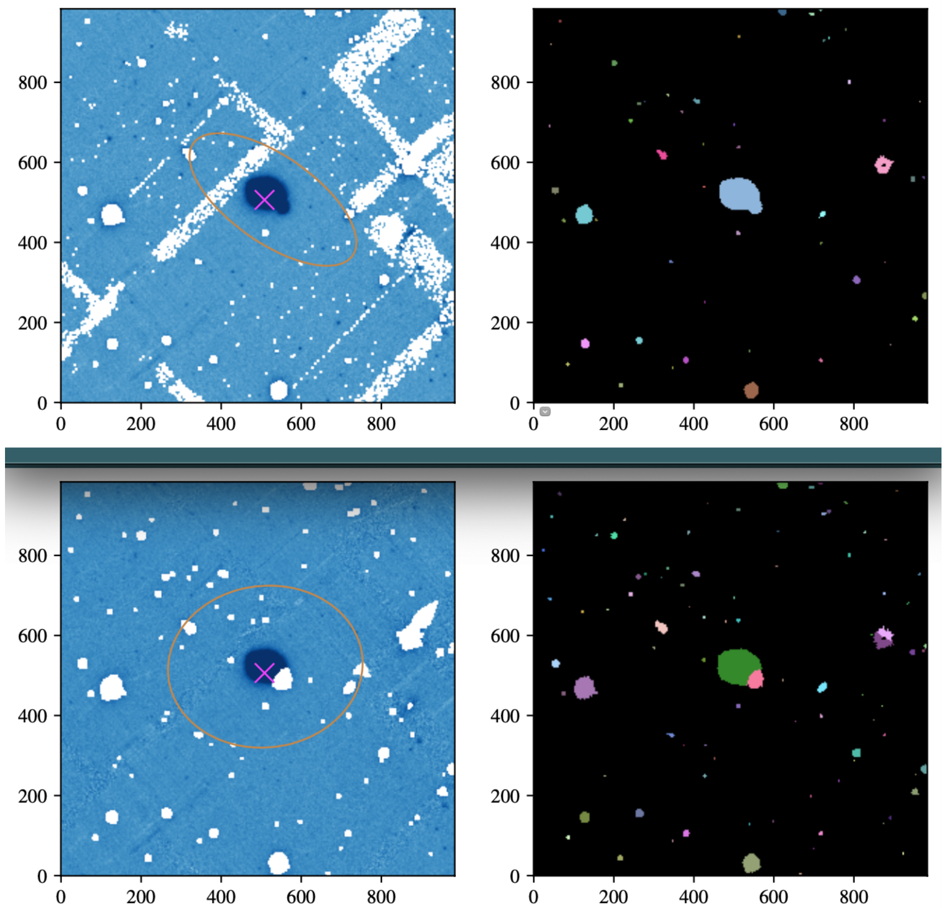

Figure 6. Top row: An example (J100746-281451) where the initial segmentation map requires deblending of the target segment. Bottom row: Result from manually setting segmentation parameters (e.g. setting a higher source finding threshold). Panels are as in Figure 4(a), top row.

measuring the photometry is to determine the local sky background, which should be subtracted prior to measuring photometry. To do this, I measure the mean flux within the outer 25% of annuli (i.e. beyond the galaxy) that also have surface brightnesses $> 26.5\,\mathrm{mag\,arcsec^{-2}}$. I then subtract this mean from the masked image and measure both the aperture and annulus photometry. The measured photometry is the total flux within each aperture/annulus in image pixel units (ADU). For the annuli, I compute the average surface brightness in ADU by dividing by the pixel area of the annuli, which is an attribute of the EllipticalAnnulus. For PanSTARRS, the image pixel units (ADU) are converted to apparent magnitudes as

$$m/\mathrm{mag} = 25 + 2.5\log(t) - 2.5\log(y), \tag{5}$$

where t is the total exposure time provided in the image header in seconds and y is the total/average ADU. The output annulus isophotes provide mean $\mathrm{ADU\,pixel^{-2}}$, which I convert to a radial surface brightness in $\mathrm{mag\,arcsec^{-2}}$ using Equation 5 and the PanSTARRS pixel size $(0.2498 \times 0.2498\,\mathrm{arcsec\,pixel^{-2}})$.

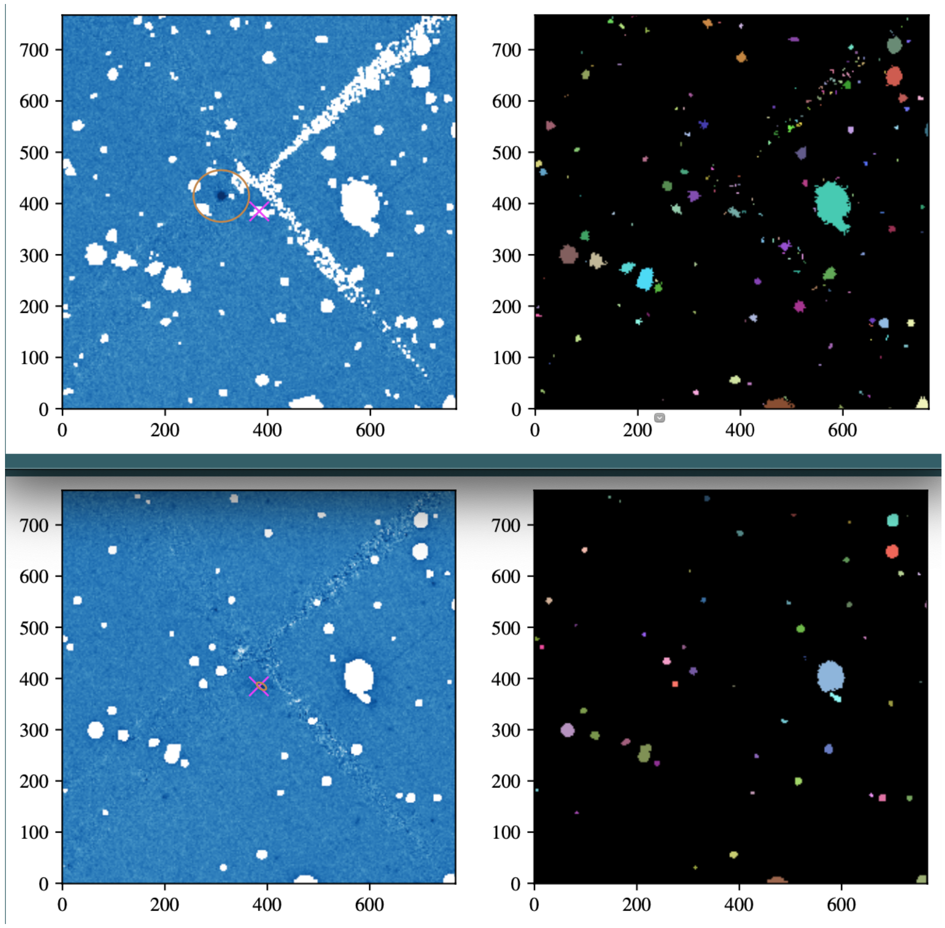

Figure 7. Top row: An example (J100555-291840) where the initial segmentation map identified an incorrect segment as the target. Bottom row: Result from manually setting segmentation parameters (e.g. setting a smaller minimum segment size). Panels are as in Figure 4(a), top row.

4.1.3 Measured Quantities

The measured annulus and aperture photometry is used to measure the optical radius (i.e. isophotal and effective) and total magnitude within an aperture defined by the given radius.

I measure the isophotal r-band radius by interpolating the r-band surface brightness profile between the radii where the surface brightness drops below the chosen isophotal limit (e.g. 25, 26 or 27 $\mathrm{mag\,arcsec^{-2}}$). The total magnitude in the g- and r-bands are then determined by interpolating the flux (ADU) curve of growth profile at the isophotal radius and then using Equation 5 to convert to apparent magnitude.

I measure the effective radius ($R_{50,\mathrm{r}}$) as the radius containing half the flux within the aperture defined by the 26 $\mathrm{mag\,arcsec^{-2}}$ isophote. To do this, I first divide the curve of growth profile (in ADU flux units) by the total flux (ADU) within the 26 $\mathrm{mag\,arcsec^{-2}}$ isophote aperture. I then interpolate this fractional curve of growth profile to determine the radius at which the profile equals 0.5. Using the fraction curve of growth profile, I can also measure the radius containing 90% of the flux ($R_{90,\mathrm{r}}$).

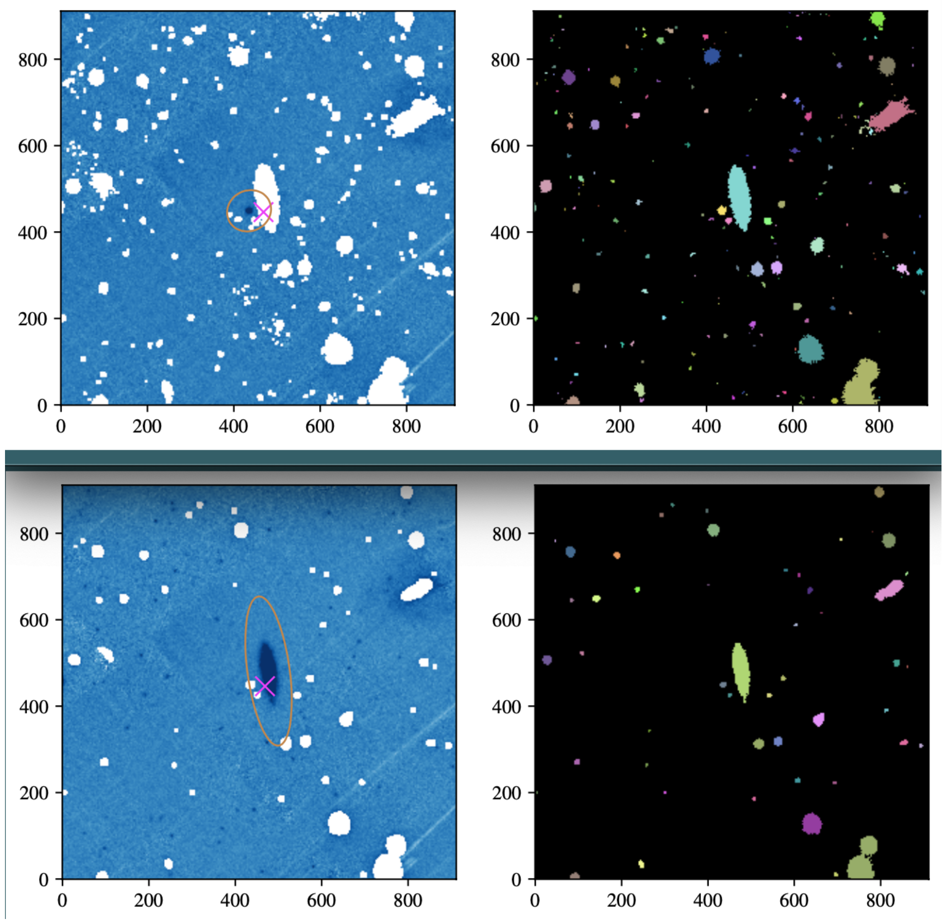

Figure 8. Top row: An example (J100720-262426) where the initial segmentation map identified an incorrect segment as the target. Bottom row: Result from manually setting segmentation parameters (e.g. setting a larger minimum segment size). Panels are as in Figure 4(a), top row.

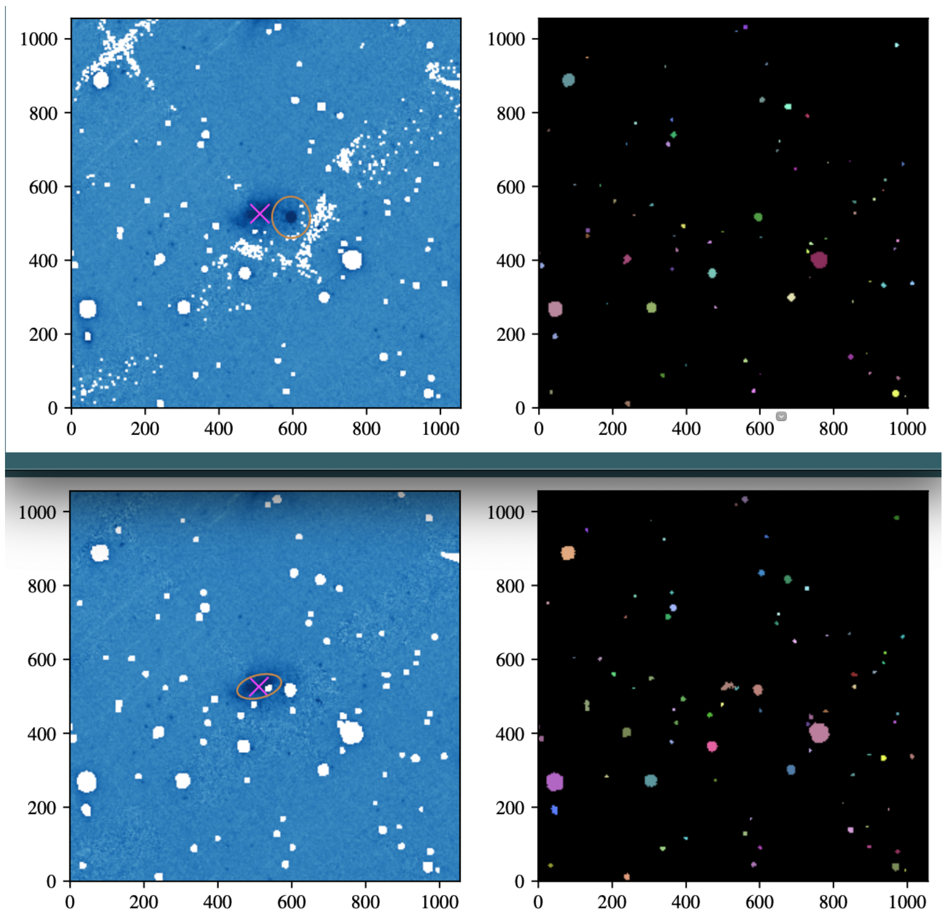

Figure 9. Top row: An example (J100808-260942) where the initial segmentation map does not find the target segment. Bottom row: Result from manually setting segmentation parameters (e.g. setting a lower source finding threshold). Panels are as in Figure 4(a), top row.

4.2 GALEX

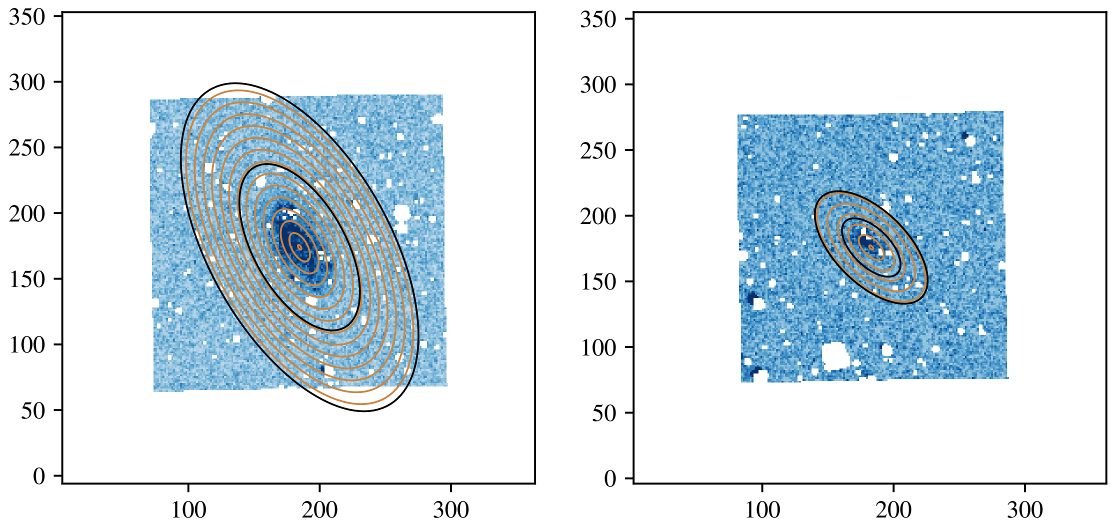

Figure 10. Example *GALEX* NUV images after applying the r-band masks and overlaying every 10th ellipse (orange) used for defining annuli and apertures. The two black ellipses indicate the range used to determine the local sky background.

The process of measuring the *GALEX* photometry is similar to that from the PanSTARRS photometry (i.e. using the PHOTUTILS package in PYTHON). I apply the PanSTARRS r-band image mask to the NUV image and then fit a series of annuli and apertures defined by the PanSTARRS segment position centre, position angle and minor-to-major axis ratio. I first reproject the PanSTARRS image mask to the *GALEX* image world coordinate system (WCS) using the PYTHON package REPROJECT and then mask all pixels in the NUV image that are masked in the r-band image mask. The *GALEX* images are larger than the PanSTARRS images, so all pixels extending beyond the PanSTARRS image footprint are also masked (e.g. Figures 10(a) and 10(b)).

I fit annuli and apertures incremented by 2 pixels out to $3 \times R_{\rm iso,r}$, where $R_{\rm iso,r}$ is the $25\,{\rm mag\,arcsec^{-2}}$ r-band radius. I determine the local mean NUV image background by averaging the annuli at radii $> 1.5R_{\rm iso,r}$ and subtract the local background. I then refit the annuli and apertures to obtain the background subtracted annulus and aperture fluxes (in image units, ADU). Figures 10(a) and 10(b) show two example galaxies with the r-band mask applied and overlaid ellipses (the ellipse colour scheme is the same as in Figure 4(b), middle row).

4.2.1 Measured Quantities

The conversion from *GALEX* image units (y) to magnitudes is

$$m/{\rm mag} = 20.08 - 2.5\log(y), \tag{6}$$

where 20.08 is the zero point magnitude in the NUV-band. Similar to the r-band, I measure the NUV-band radius by interpolating between the radii where the surface brightness drops from above to below the chosen isophote (i.e. $28\,{\rm mag\,arcsec^{-2}}$). Using the curve of growth, I measure total asymptotic magnitudes (e.g. Muñoz-Mateos et al. 2015). I perform a linear least squares fit to

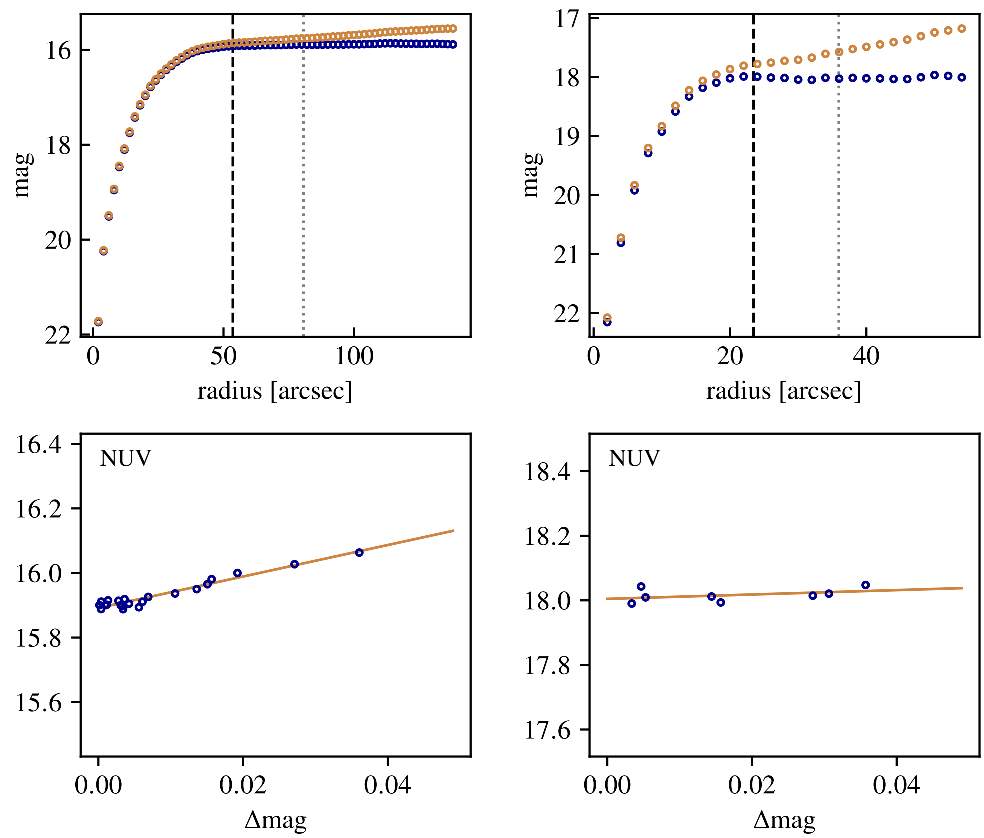

Figure 11. The extracted curve of growth profiles (top row) and the asymptotic magnitude fit to the derivative of the curve of growth (bottom row) for the two galaxies in Figures 10(a) and 10(b). In the top panels, curve of growth before (orange) and after (blue) local sky background subtraction are shown. The dashed black line shows the 28 mag arcsec^{-2} NUV radius. The dotted grey line shows the maximum radius used to determine the asymptotic magnitude.

the total magnitude vs derivative of the curve of growth, Δm, for apertures where $\Delta m < 0.05$ mag (i.e. there is little variation in the curve of growth and it is approximately flat). If there are < 4 apertures where $\Delta m < 0.05$ mag then I relax the requirement to use apertures where $\Delta m < 0.1$ mag. The total asymptotic magnitude is then found by extrapolating this linear fit to $\Delta m_{\rm fit} = 0.0$. I estimate the uncertainty in the asymptotic magnitude as the standard deviation of the magnitudes where $\Delta m < 0.05$ mag (or $\Delta m < 0.1$ mag). Figures 11(a) and 11(b) show the curve of growth and derivative of curve of growth for the two galaxies shown in Figures 10(a) and 10(b).

The APERTURE_PHOTOMETRY task estimates the error, σ_y, in the measured aperture summed ADU based on the pixel error. I calculate the source signal to noise ratio for an aperture defined using the isophotal radius (e.g. 28 mag arcsec^{-2}) as SNR = ADU$_{\rm aperture}$/ADU$_{\rm error}$, where ADU$_{\rm aperture}$ is the total aperture flux and ADU$_{\rm error}$ is the error in the aperture flux. *GALEX* magnitudes with SNR < 5 should be considered upper limits.

4.3 WISE

For the WISE bands (W1, W2, W3, W4), I apply the mask created from the r-band image to mask other objects in the WISE band images. The pixel scale of WISE is larger than that of PanSTARRS (2.75 vs 0.25 arcsec/pixel) and the WISE resolution is lower than PanSTARRS with a PSF that varies across the WISE bands (W1, W2, W3, W4: [6.08, 5.60], [6.84, 6.12], [7.36, 6.08], [11.99, 11.65] arcsec, respectively [major axis, minor axis]). For these reason, I do not measure photometry for a series of annuli/apertures as many galaxies are poorly resolved (mainly in the W3- and W4-bands). For WISE, I only measure the total magnitude for an aperture defined by the r-band 25 mag arcsec^{-2} isophotal radius.

Similar to the *GALEX* images, I reproject the PanSTARRS image mask to the WISE image world coordinate system (WCS) using the PYTHON package REPROJECT and then mask all pixels in the WISE image bands that are masked in the r-band image mask. I convolve the PanSTARRS 25 mag arcsec^{-2} aperture with the corresponding WISE band PSF prior to measuring the total magnitude within the aperture using the PHOTUTILS task APERTURE_PHOTOMETRY. The WISE images have the same zero point magnitude (22.5) and the image units (y) are converted to magnitudes using

$$m/\mathrm{mag} = 22.5 - 2.5 \log(y). \tag{7}$$

If a galaxy is well resolved then the convolved aperture is very similar to the original PanSTARRS aperture (e.g. Figure 12(a)). If a galaxy is less resolved then the convolved aperture can be significantly larger than the original PanSTARRS aperture (e.g. Figure 12(b) and 12(c)).

The APERTURE_PHOTOMETRY task estimates the error, σ_y, in the measured aperture summed ADU based on the pixel error. I estimate the pixel error as the standard deviation of an annulus around the source aperture with the same position and inclination angles and the size defined by $a_{\mathrm{in}} = 1.5a$, $a_{\mathrm{out}} = 2.5a$, $b_{\mathrm{in}} = 1.5b$ and $b_{\mathrm{out}} = 2.5b$, where a, b are the aperture major and minor axes. The source signal to noise ratio is then $\mathrm{SNR} = \mathrm{ADU}_{\mathrm{aperture}}/\mathrm{ADU}_{\mathrm{error}}$. WISE magnitudes with $\mathrm{SNR} < 5$ should be considered upper limits (e.g. $\mathrm{SNR} = 1.4$ and -4.7 for the W3- and W4-bands in the right two bottom row panels of Figure 12(c)).

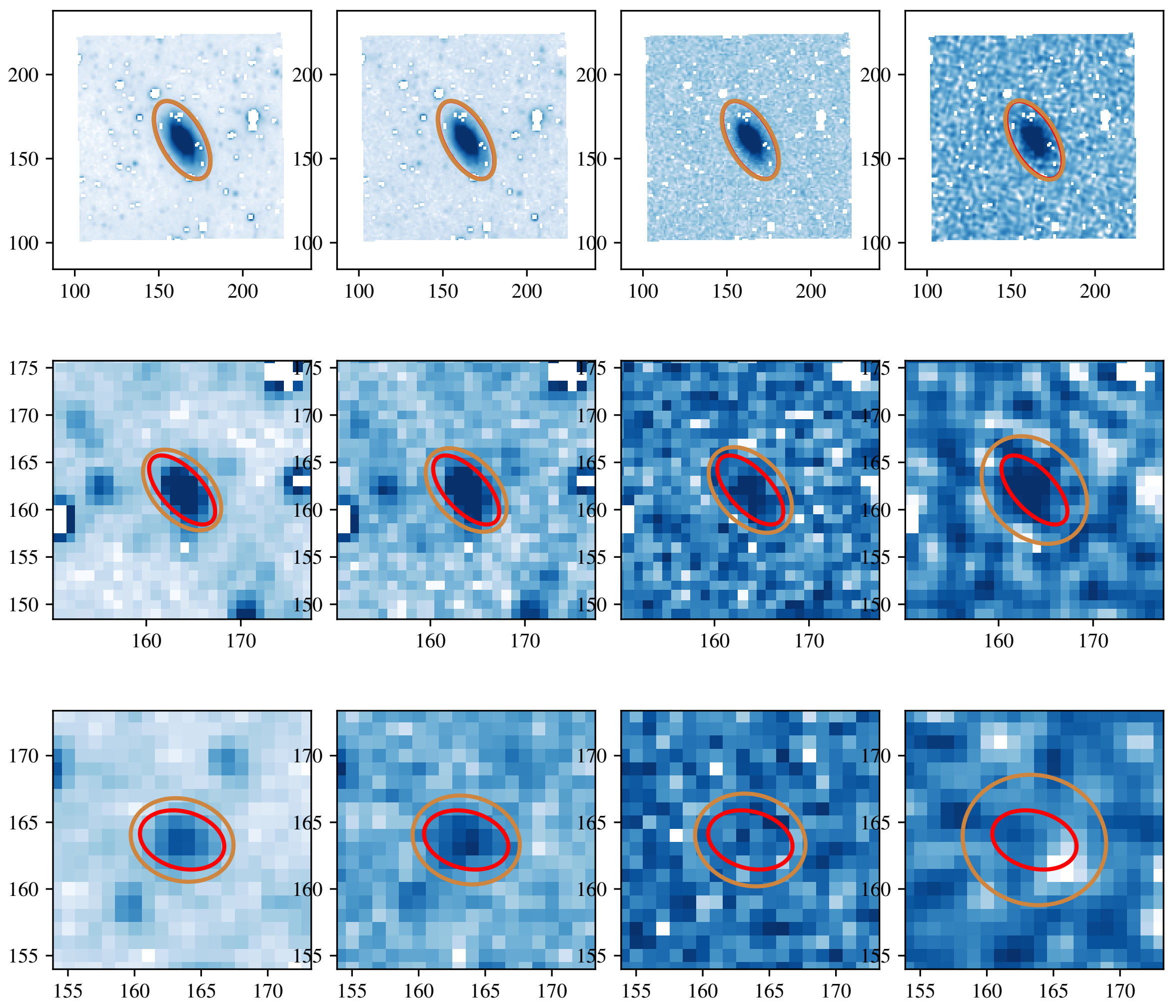

Figure 12. Example WISE band images (W1, W2, W3, W4 from left to right) with overlaid ellipses showing the PanSTARRS aperture (red) and convolved PanSTARRS aperture (orange). For well resolved galaxies, the apertures are nearly identical (Figure 12(a)). For less resolved galaxies, the convolved aperture is significantly larger than the original PanSTARRS aperture with the largest difference being for the W4 band (Figure 12(b) and 12(c)).

5 Derived Quantities

5.1 Stellar Mass

The PanSTARRS g- and r-band image photometry are used to derive stellar masses. I use the empirical relation from Taylor et al. (2011) and PanSTARRS g- and r-band magnitudes measured within an aperture defined by the chosen isophotal radius (e.g. 25 mag arcsec^{-2}) to estimate stellar masses,

$$\log\left(\frac{M_*}{\mathrm{M}_\odot}\right) = -0.840 + 1.654(g-r) + 0.4(D_{\mathrm{mod}} + M_{\mathrm{sol}} - r) - \log(1+z) - 2\log\left(\frac{h}{0.7}\right), \qquad (8)$$

where -0.840 and 1.654 are empirically determined constants based on the r-band magnitude and $g-r$ colour from Zibetti et al. (2009), the $g-r$ colour is in the SDSS photometric system, r is the r-band apparent magnitude in the SDSS photometric system, D_{mod} is the distance modulus (used to convert from apparent to absolute magnitude) and $M_{\mathrm{sol}} = 4.64$ is the absolute magnitude of the Sun in the r-band (Willmer 2018). I convert magnitudes and colours from PanSTARRS to the SDSS photometric systems using equation 6 from Tonry et al. (2012),

$$g_{\mathrm{sdss}} = 0.014 + 0.162 * (g_{\mathrm{ps}} - r_{\mathrm{ps}}) + g_{\mathrm{ps}} \qquad (9)$$

and

$$r_{\mathrm{sdss}} = 0.014 + 0.162 * (g_{\mathrm{ps}} - r_{\mathrm{ps}}) + r_{\mathrm{ps}}, \qquad (10)$$

where g_{ps} and r_{ps} are the PanSTARRS photometric system magnitudes and g_{sdss} and r_{sdss} are the SDSS photometric system magnitudes.

5.2 Galactic Dust Extinction

I correct for Galactic extinction in the g- and r-bands. Assuming the dust extinction law of Cardelli et al. (1989), the Galactic dust attenuation for a given galaxy is approximated by $A_{\mathrm{v}} = R_V E(B-V)$, where $R_V = 3.793$ and 2.751 for the g- and r-bands, respectively (Wyder et al. 2007). The reddening for a given galaxy, $E(B-V)$, is obtained using the PYTHON module ASTROQUERY to query the IRSA Dust Extinction Service[6] at the sky position of each H I detection. This provides two estimates of the dust extinction from Schlegel et al. (1998) and Schlafly & Finkbeiner (2011). I use the Schlegel et al. (1998) values. Magnitudes corrected for Galactic dust absorption are then

$$m_{\mathrm{cor}} = m_{\mathrm{sdss}} - A_{\mathrm{v}}. \qquad (11)$$

5.3 Star Formation Rate

I calculate total star formation rates (SFRs) by combining the contributions derived from *GALEX* NUV and WISE mid-infrared (MIR) magnitudes following Janowiecki et al. (2017). I first show the conversion from measured *GALEX* and WISE magnitudes to luminosities, which are used to derived SFRs.

[6] https://irsa.ipac.caltech.edu/applications/DUST/index.html

5.3.1 Converting Flux to Luminosity

GALEX I correct *GALEX* NUV magnitudes for Galactic extinction following the same method as the PanSTARRS g- and r-bands, with $R_V = 8.2$ from Wyder et al. (2007) for the NUV. The conversion from *GALEX* flux (f_{G}) to luminosity (L_{G}) is given by

$$\frac{L_{\mathrm{G}}}{\mathrm{erg\,s^{-1}\,Hz^{-1}}} = \left(\frac{f_{\mathrm{G}}}{\mathrm{erg\,s^{-1}\,cm^{-2}\,\AA^{-1}}}\right)\left(\frac{A}{\mathrm{cm^2}}\right)\left(\frac{\nu}{\mathrm{Hz}}\right)\left(\frac{\mathrm{\AA}}{\lambda}\right), \tag{12}$$

where A is the total surface area of a sphere at the observer's distance, ν is the observed frequency and $\lambda = 2315.7,\ 1538.6\,\mathrm{\AA}$ is the observed wavelength in angstroms (NUV, FUV; respectively). The conversion from magnitude to flux is given by

$$f_{\mathrm{G}} = f_0 10^{-(m-m_0)/2.5} \tag{13}$$

where $f_0 = 2.06 \times 10^{-16},\ 1.40 \times 10^{-15}\,\mathrm{erg\,s^{-1}\,cm^{-2}\,Hz^{-1}}$ and $m_0 = 20.08,\ 18.82\,\mathrm{mag}$ are *GALEX* band-specific constants (NUV, FUV; respectively) and m is the measured magnitude. The surface area of the sphere is given by

$$\frac{A}{\mathrm{cm^2}} = 4\pi\left(\frac{d_{\mathrm{L}}}{\mathrm{Mpc}}\right)^2\left(\frac{3.086 \times 10^{24}}{\mathrm{cm}}\right)^2, \tag{14}$$

where d_{L} is the luminosity distance to the source and $1\,\mathrm{Mpc} = 3.086 \times 10^{24}\,\mathrm{cm}$.

WISE The conversion from WISE flux (f_{W}) to luminosity (L_{W}) is given by

$$\frac{L_{\mathrm{W}}}{\mathrm{L_\odot}} = \left(\frac{f_{\mathrm{W}}}{\mathrm{Jy}}\right)\left(\frac{A}{\mathrm{cm^2}}\right)\left(\frac{\nu}{\mathrm{Hz}}\right)\left(\frac{10^{-23}}{\mathrm{erg\,s^{-1}\,cm^{-2}\,Hz^{-1}}}\right)\left(\frac{\mathrm{erg\,s^{-1}}}{3.828 \times 10^{33}}\right) \tag{15}$$

where A is the total surface area of a sphere at the observer's distance, ν is the observed frequency, $1\,\mathrm{Jy} = 10^{-23}\,\mathrm{erg\,s^{-1}\,cm^{-2}\,Hz^{-1}}$ and $1\,\mathrm{L_\odot} = 3.828 \times 10^{33}\,\mathrm{erg\,s^{-1}}$. The conversion from magnitude to flux is given by

$$f_{\mathrm{W}} = f_0 10^{-m/2.5} \tag{16}$$

where $f_0 = 309.540,\ 171.787,\ 31.674,\ 8.363\,\mathrm{Jy}$ is a WISE band-specific constant (W1, W2, W3, W4; respectively) and m is the measured magnitude. The frequencies are calculated from the observed wavelength ($\lambda = 3.4,\ 4.6,\ 12,\ 22\,\mu\mathrm{m}$ for W1, W2, W3, W4; respectively). The surface area of the sphere is given by Equation 14.

5.3.2 SFR

Unobscured SFR The UV SFR is derived from the *GALEX* NUV band luminosity, L_{NUV}, following Schiminovich et al. (2007) as,

$$\frac{\mathrm{SFR_{NUV}}}{\mathrm{M_\odot yr^{-1}}} = 10^{-28.165}\left(\frac{L_{\mathrm{NUV}}}{\mathrm{erg\,s^{-1}\,Hz^{-1}}}\right). \tag{17}$$

Obscured SFR The MIR SFR comes from the WISE W4 band luminosity, L_{W4}. If a galaxy is undetected in W4, then the W3 band luminosity, L_{W3}, is used instead. The W3 and W4 WISE bands contain contamination from older stellar populations and require a correction by subtracting a fraction of the W1 band luminosity, L_{W1} (Ciesla et al. 2014). The W3 and W4 derived SFRs are given by (Jarrett et al. 2013)

$$\frac{\mathrm{SFR}_{\mathrm{W3}}}{\mathrm{M}_{\odot}\mathrm{yr}^{-1}} = 4.91 \times 10^{-10} \left(\frac{L_{\mathrm{W3}} - 0.201 L_{\mathrm{W1}}}{\mathrm{L}_{\odot}} \right) \tag{18}$$

and

$$\frac{\mathrm{SFR}_{\mathrm{W4}}}{\mathrm{M}_{\odot}\mathrm{yr}^{-1}} = 7.50 \times 10^{-10} \left(\frac{L_{\mathrm{W4}} - 0.044 L_{\mathrm{W1}}}{\mathrm{L}_{\odot}} \right). \tag{19}$$

If the W3 or W4 luminosities corrected for old stellar contamination are negative (i.e. $L_{\mathrm{W3}} - 0.201 L_{\mathrm{W1}} < 0$ or $L_{\mathrm{W4}} - 0.044 L_{\mathrm{W1}} < 0$), then I use the uncorrected W3 or W4 luminosities and flag the resulting MIR SFR as an upper limit. I note that there are more recent WISE-based SFR calibrations from Cluver et al. (2017), however I use the Jarrett et al. (2013) calibrations to enable comparison with results from xGASS (Catinella et al. 2018; Janowiecki et al. 2017, 2020).

Total SFR The total SFR is then

$$\mathrm{SFR}_{\mathrm{NUV+MIR}} = \mathrm{SFR}_{\mathrm{NUV}} + \mathrm{SFR}_{\mathrm{W4(3)}}. \tag{20}$$

The derived SFRs have uncertainties of $\lesssim 0.1\,\mathrm{dex}$.

6 Analysis Steps

The following are the steps involved in measuring multi-wavelength photometry for WALLABY detections in a team release (e.g. Hydra DR2).

1. Manually create base directory path where all data will be located. (**NOTE: `basedir` will need to be updated in all analysis scripts to your directory path.**)
   - E.g. `basedir` = `/Users/tflowers/WALLABY/PHASE1/Hydra_DR2/`
   - `/Users/tflowers/WALLABY/` is the parent directory path that will contain all WALLABY data
   - `PHASE1/Hydra_DR2/` is the specific directory path for a particular survey phase (e.g. `PHASE1`, `PHASE2`) and team release (e.g. `Hydra_DR2`, `NGC4636_DR1`, `NGC4808_DR1`, etc.)
2. Before running any PYTHON scripts update the following (see Appendix A.1):
   - `basedir` to the base directory specified in step 1. Only update the parent directory path segment (e.g. `/Users/tflowers/WALLABY/`) as the phase/team release segment is dynamically set by the variables `survey_phase` and `team_release`.
   - Add the survey phase and team release to the `survey_phase_list` and `team_release_list` arrays and set `tr_i` to the corresponding index (NOTE: PYTHON indexing starts at 0). This sets the variables `survey_phase` and `team_release`.
3. Prepare WALLABY data
   - Download WALLABY data (SoFiA source catalogue and data products) from the WALLABY database.
   - Create and move WALLABY data to directory `basedir` + `SOFIA/`
   - Rename SoFiA source catalogue as `*_catalogue.fits`, where `*` = team release (e.g. `Hydra_DR2`)
   - Rename SoFiA data products directory as `*_source_products`, where `*` = team release
   - If the team release name is `NGC*`, remove the underscore between NGC and the number (e.g. change `NGC_5044` to `NGC5044`)
4. Run `create_directories.py` to create full directory trees (see Appendix A.2)
5. Set the paths to the *GALEX* and WISE image tables (`galex_images.csv.gz` and `unwise_tiles.fits.gz`) in the `download_galex.py` and `download_wise.py` scripts (Sections 3.2 and 3.3). These files are located in the same directory as the PYTHON scripts.
   - *GALEX*: In the `get_galex_images(ra, dec, size)` function update the `galex_image_table` variable (e.g. `galex_image_table = '/Users/tflowers/CODE/wallaby/galex_images.csv.gz'`)
   - WISE: In the `get_unwise_tiles(ra, dec, size)` function update the `wise_tile_table` variable (e.g. `wise_tile_table = '/Users/tflowers/CODE/wallaby/unwise_tiles.fits.gz'`)

6. Download PanSTARRS/*GALEX*/WISE image cutouts by running `download_*.py`, where * is `panstarrs`, `galex`, `wise` (see Section 3 and Appendix A.3). This will take several hours, so leave running in the background (depending on total number of H I detections). The switch settings are shown below:
    - `*panstarrs.py`: `open_catalogue=True`, `do_download_panstarrs=True`
    - `*galex.py`: `open_catalogue=True`, `do_download_galex=True`
    - `*wise.py`: `open_catalogue=True`, `do_download_wise=True`
7. Create SoFiA catalogue table with flag columns by running `photometry_wallaby.py` (see Section 1.1):
    - Switch settings: `do_get_source_properties=True`, `have_optical=False`, `table_add_flags=True`, `do_measure_hi=False`, `do_hi_opt_disc=False`
    - Output: `basedir + SOFIA/*_catalogue_flags.fits`, where `*` = team release
8. Measure H I structural parameters by running `photometry_wallaby.py` (see Section 2.2.2 and Appendix A.4):
    - Switch settings: `do_get_source_properties=True`, `have_optical=False`, `table_add_flags=False`, `do_measure_hi=True`, `do_hi_opt_disc=False`
    - Output: `basedir + HI_DERIVED_PRODUCTS/PROFILES/*_profiles.fits`, where `*` = galaxy name
    - Output: `basedir + PARAMETERS/*_hi_structural_parameters.fits`, where `*` = team release
9. Once the PanSTARRS images have finished downloading, can run `galaxy_summary_plots.py` to create plots showing PanSTARRS r-band image with overlaid H I moment 0 map contours, H I moment 1 map and integrated H I spectrum.
    - Switch settings: `open_catalogue=True`, `do_plot_maps=True`
    - Output: PDF images in `basedir + PLOTS/MAPS/ALL/*_maps.pdf`, where `*` = galaxy name
10. Inspect summary plots to identify issues with data (e.g. artefacts, multiple counterparts, shredded H I source, foreground star) and update source classification flags (`flag_src_class` and `flag_opt_fit`) in `*_catalogue_flags.fits` accordingly (e.g. using TOPCAT; see Section 1.1).
11. Run `photometry_panstarrs.py` to create initial PanSTARRS r-band segmentation maps and define target segment (see Section 4.1.1 and Appendix A.6.1).
    - Switch settings: `do_get_source_properties=True`, `have_segments=False`, `do_segim=True`, `do_segim_bespoke=False`, `do_fit_phot=False`, `do_measure=False`
    - Output: PDF images in `basedir + PLOTS/PHOTOMETRY/PANSTARRS/OPTICAL_SEGMENTATION/*_segmap.pdf`, where `*` = galaxy name
    - Output: `basedir + PARAMETERS/*_panstarrs_segments.fits`, where `*` = team release

12. Run photometry_panstarrs.py to fix segmentation maps/target segments for sources that failed during the initial run (see Section 4.1.1 and Appendix A.6.1). If there is no output file named *_panstarrs_segments_deblend.fits, then *_panstarrs_segments.fits is copied to *_panstarrs_segments_deblend.fits. Each time the script is run for a specified galaxy, the segment parameters for that galaxy are updated in the *_panstarrs_segments_deblend.fits file, which is backed-up to *_panstarrs_segments_deblend_old.fits.
    - Switch settings: do_get_source_properties=True, have_segments=True, do_segim=False, do_segim_bespoke=True, do_fit_phot=False, do_measure=False
    - Output: PDF images in basedir + PLOTS/PHOTOMETRY/PANSTARRS/OPTICAL_SEGMENTATION/*_segmap.pdf, where * = galaxy name
    - Output: basedir + PARAMETERS/*_panstarrs_segments_deblend.fits, where * = team release
    - Once the segmentation map is fixed, copy the segment/deblend parameters into the *_catalogue_flags.fits file (e.g. do_seg_par* and do_deblend*).
    - Alternatively, if the segmentation map cannot be fixed, then update the optical flag in *_catalogue_flags.fits to flag_opt_fit = 5 (i.e. failed due to image artefacts).
13. Once finished fixing the segmentation maps/target segments, run photometry_panstarrs.py to fit annuli/apertures to masked g- and r-band images (see Section 4.1.2 and Appendix A.6.1).
    - Switch settings: do_get_source_properties=True, have_segments=True, do_segim=False, do_segim_bespoke=False, do_fit_phot=True, do_measure=False
    - Output: PDF images in basedir + PLOTS/PHOTOMETRY/PANSTARRS/MAP_ELLIPSE/*_map_ellipse.pdf, where * = galaxy name
    - Output: basedir + MULTIWAVELENGTH/PANSTARRS/PROFILES_BKGDSUB/*_panstarrs_segments.fits, where * = galaxy name
14. Run photometry_panstarrs.py to measure g- and r-band photometry and optical disc sizes (see Section 4.1.3 and Appendix A.6.1).
    - Switch settings: do_get_source_properties=True, have_segments=True, do_segim=False, do_segim_bespoke=False, do_fit_phot=False, do_measure=True
    - Output: PDF images in basedir + PLOTS/PHOTOMETRY/PANSTARRS/OPTICAL_PROFILES/*_profiles.pdf, where * = galaxy name
    - Output: basedir + PARAMETERS/*_panstarrs_photometry.fits, where * = team release
15. Run photometry_galex.py to fit annuli/apertures to the NUV-band image (see Section 4.2 and Appendix A.6.2).
    - Switch settings: do_get_source_properties=True, have_segments=True, have_optical=False, do_fit_phot=True, do_measure=False
    - Output: PDF images in basedir + PLOTS/PHOTOMETRY/GALEX/APERTURES/*_aperture.pdf, where * = galaxy name

- Output: `basedir + MULTIWAVELENGTH/GALEX/PROFILES/*_profile.fits`, where `*` = galaxy name

16. Run `photometry_galex.py` to measure NUV-band photometry and disc size (see Section 4.2.1 and Appendix A.6.2).
    - Switch settings: `do_get_source_properties=True`, `have_segments=True`, `have_optical=False`, `do_fit_phot=False`, `do_measure=True`
    - Output: PDF images in `basedir + PLOTS/PHOTOMETRY/GALEX/ASYMPTOTIC_MAG/*_asymptote_fit.pdf`, where `*` = galaxy name
    - Output: `basedir + PARAMETERS/*_galex_photometry.fits`, where `*` = team release
17. Run `photometry_wise.py` to measure WISE photometry (see Section 4.3 and Appendix A.6.3)
    - Switch settings: `do_get_source_properties=True`, `have_segments=True`, `have_optical=True`, `do_measure=True`
    - Output: PDF images in `basedir + PLOTS/PHOTOMETRY/WISE/*_aperture.pdf`, where `*` = galaxy name
    - Output: `basedir + PARAMETERS/*_wise_photometry.fits`, where `*` = team release
18. Measure H I within the optical disc by running `photometry_wallaby.py` (see Section 2.2.3 and Appendix A.4):
    - Switch settings: `do_get_source_properties=True`, `have_optical=True`, `table_add_flags=False`, `do_measure_hi=False`, `do_hi_opt_disc=True`
    - Output: `basedir + PARAMETERS/*_hi_optical_disc.fits`, where `*` = team release
19. Create table with Galactic dust extinction values, $E(B-V)$, by running `derived_properties.py` (see Section 5.2 and Appendix A.7.1):
    - Switch settings: `do_open_tables=True`, `do_get_source_properties=True`, `do_get_dust_extinction=True`, `do_derive_quantities=False`, `do_pilot_survey_all=False`
    - Output: `basedir + PARAMETERS/*_galactic_dust_extinction.fits`, where `*` = team release
20. Derive physical galaxy quantities (e.g. stellar and H I masses and SFRs) by running `derived_properties.py` (see Section 5):
    - Switch settings: `do_open_tables=True`, `do_get_source_properties=True`, `do_get_dust_extinction=False`, `do_derive_quantities=True`, `do_pilot_survey_all=False`
    - Output: `basedir + PARAMETERS/*_derived_galaxy_properties.fits`, where `*` = team release
    - Output: `basedir + PARAMETERS/*_derived_galaxy_sfrs.fits`, where `*` = team release

A Python Scripts

A.1 Basic Inputs/Controls

Basic inputs required by all scripts to specify the WALLABY team release to use and specifying directory paths and controls specifying which tasks to run. The examples shown under `Switches` are from `photometry_wallaby.py` and are specific to each PYTHON script.

```
# =================================== #
# =========== Switches ============ #
# =================================== #
do_get_source_properties  = True       # Always True, provides input source parameters
have_optical              = True       # True to open *_panstarrs_photometry.fits if  exists

# ++++ ONLY RUN ONE AT A TIME +++++ #
table_add_flags           = False      # Only set True once to create SoFiA catalogue with flag columns

do_measure_hi             = False      # True to measure HI structural properties
do_hi_opt_disc            = True       # True to measure HI mass/surface density w/in optical disc

# =================================== #
# ==== Specify Phase + Release ==== #
# =================================== #
tr_i                      = 1
survey_phase_list         = ['PHASE1', 'PHASE1', 'PHASE1',
                             'PHASE2', 'PHASE2', 'PHASE2']
team_release_list         = ['Hydra_DR1', 'Hydra_DR2', 'NGC4636_DR1',
                             'NGC4808_DR1', 'NGC5044_DR1', 'NGC5044_DR2']
survey_phase              = survey_phase_list [ tr_i ]
team_release              = team_release_list [ tr_i ]

# =================================== #
# ========= File Strings ========== #
# =================================== #
basedir                   = '/Users/tflowers/WALLABY/%s/%s/' % (survey_phase, team_release)
sofia_dir                 = basedir + 'SOFIA/'
dataprod_dir              = basedir + 'SOFIA/%s_source_products/' % team_release
panstarrs_dir             = basedir + 'MULTIWAVELENGTH/PANSTARRS/'
wise_dir                  = basedir + 'MULTIWAVELENGTH/WISE/'
galex_dir                 = basedir + 'MULTIWAVELENGTH/GALEX/'
parameter_dir             = basedir + 'PARAMETERS/'
hi_products_dir           = basedir + 'HI_DERIVED_PRODUCTS/'
plots_dir                 = basedir + 'PLOTS/'
```

A.2 Create Directory Structure

`create_directories.py`

```
create_directories  = True
if  create_directories :
  print('============ Make Directories ============')
  os.system('mkdir %s' % (basedir + 'MULTIWAVELENGTH/'))
  os.system('mkdir %s' % panstarrs_dir)
  os.system('mkdir %s' % (panstarrs_dir + 'PROFILES_BKGDSUB/'))
  os.system('mkdir %s' % wise_dir)
```

```
os.system('mkdir %s' % (wise_dir + 'PROFILES/'))
os.system('mkdir %s' % galex_dir)
os.system('mkdir %s' % (galex_dir + 'PROFILES/'))
os.system('mkdir %s' % parameter_dir)
os.system('mkdir %s' % hi_products_dir)
os.system('mkdir %s' % (hi_products_dir + 'PROFILES/'))
os.system('mkdir %s' % plots_dir)
os.system('mkdir %s' % (plots_dir + 'MAPS/'))
os.system('mkdir %s' % (plots_dir + 'MAPS/ALL/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/HI_PROFILES/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/PANSTARRS/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/PANSTARRS/MAP_ELLIPSE/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/PANSTARRS/OPTICAL_PROFILES/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/PANSTARRS/OPTICAL_SEGMENTATION'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/GALEX/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/GALEX/APERTURES/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/GALEX/ASYMPTOTIC_MAG/'))
os.system('mkdir %s' % (plots_dir + 'PHOTOMETRY/WISE/'))
```

The assumed file directory structure, which is created by the `create_directories.py` script is shown below (square brackets indicate directories that the user names):

```
[BASE DIRECTORY] (e.g. WALLABY)
└── [SURVEY PHASE] (e.g. PHASE1)
    └── [TEAM RELEASE] (e.g. HYDRA_DR2)
        ├── SOFIA
        │   └── [TEAM RELEASE]_source_products
        ├── MULTIWAVELENGTH
        │   ├── PANSTARRS
        │   │   └── PROFILES_BKGDSUB
        │   ├── GALEX
        │   │   └── PROFILES
        │   ├── WISE
        │   │   └── PROFILES
        │   └── DECAM
        │       └── PROFILES
        ├── PARAMETERS
        ├── PLOTS
        │   ├── MAPS
        │   │   └── ALL
        │   └── PHOTOMETRY
        │       ├── GALEX
        │       │   ├── APERTURES
        │       │   └── ASYMPTOTIC_MAG
        │       ├── PANSTARRS
        │       │   ├── OPTICAL_SEGMENTATION
        │       │   ├── MAP_ELLIPSE
        │       │   └── OPTICAL_PROFILES
        │       ├── WISE
        │       └── HI_PROFILES
        └── HI_DERIVED_PRODUCTS
```

```
└── PROFILES
```

A.3 Downloading Data

The required inputs to make PanSTARRS, *GALEX* or WISE image cutouts are the galaxy name and SoFiA RA and Dec, which come from the SoFiA source catalogue.

```
if open_catalogue:
  print('============= %s =============' % team_release)
  fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release

  # ======== WALLABY PARAMETERS ======== #
  hdu_sofia       = fits.open( fits_sofia )
  data_sofia      = hdu_sofia[1].data

  gal_name = []
  for i in range(len(data_sofia['name'])):
    split_name = data_sofia['name'][i][8:]
    gal_name.append(split_name)

  galaxies        = np.array(gal_name)          # Galaxy name (e.g. J104059-270456)
  sofia_ra        = data_sofia['ra']            # HI detection RA
  sofia_dec       = data_sofia['dec']           # HI detection Dec
```

A.3.1 PanSTARRS Data

`download_panstarrs.py`

```
def geturl(ra, dec, size=240, output_size=None, filters="grizy", format="jpg", color=False):
    """Get URL for images in the table

    ra, dec = position in degrees
    size = extracted image size in pixels (0.25 arcsec/pixel)
    output_size = output (display) image size in pixels (default = size).
                  output_size has no effect for fits format images.
    filters = string with filters to include
    format = data format (options are "jpg", "png" or "fits")
    color = if True, creates a color image (only for jpg or png format).
            Default is return a list of URLs for single-filter grayscale images.
    Returns a string with the URL
    """

fits_tan   = galaxies[i] + '_' + bands[j] +'.fits'
fdownload = panstarrs_gal_dir + fits_tan
fitsurl    = geturl( sofia_ra[i], sofia_dec[i], size=size, filters=bands[j], format="fits")
wget.download(fitsurl[0], fdownload)
```

A.3.2 GALEX Data

`download_galex.py`

```
def get_galex_image(ra, dec, size, outname, tmpdir=None, clean=True):
    '''
```

```
    ra:      Cutout centre position RA
    dec:     Cutout centre position Dec
    size:    Cutout image size [degrees]
    outname: Cutout file name, to which nuv or fuv will be appended
    tmpdir:  Directory to hold temporary full GALEX tiles used to create cutout
    clean:   Whether to delete files from tmpdir

    Produces GALEX image cutout
    '''

galex_gal_dir  = galex_dir + galaxies[i] + '/'
tmp_dir        = galex_gal_dir + 'tmpdir'

base_name      = galex_gal_dir + galaxies[i]
get_galex_image(sofia_ra[i], sofia_dec[i], 0.15, base_name, clean=False, tmpdir=tmp_dir)
os.system('rm -rf %s' % tmp_dir)
```

A.3.3 WISE Data

`download_wise.py`

```
def get_unwise_image(ra, dec, size, outname, tmpdir=None, clean=True):
    '''
    ra:      Cutout centre position RA
    dec:     Cutout centre position Dec
    size:    Cutout image size [degrees]
    outname: Cutout file name, to which nuv or fuv will be appended
    tmpdir:  Directory to hold temporary full GALEX tiles used to create cutout
    clean:   Whether to delete files from tmpdir

    Produces GALEX image cutout
    '''

wise_gal_dir   = wise_dir + galaxies[i] + '/'
tmp_dir        = wise_gal_dir + 'tmpdir'

base_name      = wise_gal_dir + galaxies[i]
get_unwise_image(sofia_ra[i], sofia_dec[i], 0.25, base_name, clean=False, tmpdir=tmp_dir)
os.system('rm -rf %s' % tmp_dir)
```

A.4 WALLABY Measurements

The required inputs to measure H I properties from the SoFiA moment 0 maps.

```
# ================================= #
# ===== Open/Join FITS Tables ===== #
# ================================= #
if ~have_optical:
  print('============ %s ============' % team_release)
  fits_sofia     = sofia_dir + '%s_catalogue.fits' % team_release

  hdu_sofia      = fits.open(fits_sofia)
  data_join      = hdu_sofia[1].data
if have_optical:
  print('============ %s ============' % team_release)
```

```
    fits_sofia        = sofia_dir  + '%s_catalogue.fits' % team_release
    fits_panstarrs    = parameter_dir + '%s_panstarrs_photometry.fits' % team_release

    hdu_sofia         = fits.open( fits_sofia )
    data_sofia        = hdu_sofia[1].data

    hdu_panstarrs     = fits.open( fits_panstarrs )
    data_panstarrs    = hdu_panstarrs[1].data

    data_join         = join(data_sofia, data_panstarrs, join_type='left')

# =================================== #
# ===== Get Source Properties ===== #
# =================================== #
if do_get_source_properties:
    #flag_mask        = ((data_join['flag_class'] == 1) | (data_join['flag_class'] == 4))
    data_mask        = data_join#[flag_mask]
    gal_name = []
    for i in range(len(data_mask['name'])):
      split_name = data_mask['name'][i][8:]
      gal_name.append(split_name)

    galaxy_dir  = 'WALLABY_' + gal_name[0] + '/'
    fits_file   = dataprod_dir + galaxy_dir + 'WALLABY_' + gal_name[0] + '_cube.fits.gz'

    f1          = pyfits.open( fits_file )
    data, hdr   = f1[0].data, f1[0].header
    if hdr['CTYPE3'] == 'FREQ':
      chan_width = np.abs((HI_REST / (hdr['CRVAL3']/1e6)) - (HI_REST /
            ((hdr['CRVAL3']-hdr['CDELT3'])/1e6))) * C_LIGHT
      chan_width_hz = hdr['CDELT3']
    else:
      chan_width = np.abs(hdr['CDELT3']/1000.)
    beam_maj, beam_min, beam_pa, pix_scale = hdr['BMAJ'], hdr['BMIN'], hdr['BPA'], np.abs(hdr['CDELT1'])
    f1.close()

    BEAM          = beam_factor(beam_maj*3600., beam_min*3600., pix_scale*3600.)

    print(BEAM)

    redshift      = (HI_REST / (data_mask['freq'] / 1.e6)) - 1.
    galaxies      = np.array(gal_name)
    sofia_ra      = data_mask['ra']
    sofia_dec     = data_mask['dec']
    sofia_vsys    = redshift * C_LIGHT
    sofia_rms     = data_mask['rms']
    sofia_sint    = data_mask['f_sum'] * chan_width / chan_width_hz
    sofia_snr     = data_mask['f_sum'] / data_mask['err_f_sum']
    sofia_kinpa   = data_mask['kin_pa']
    sofia_w20     = data_mask['w20'] / chan_width_hz * chan_width
    sofia_w50     = data_mask['w50'] / chan_width_hz * chan_width
    sofia_ell_maj = data_mask['ell_maj'] * pix_scale * 3600.
    sofia_ell_min = data_mask['ell_min'] * pix_scale * 3600.
    sofia_ellpa   = data_mask['ell_pa']

    #Correction for measured WALLABY fluxes
    sint_corr     = np.array(wallaby_flux_scaling(data_mask['f_sum']))
    scale_factor  = np.array(data_mask['f_sum'] / sint_corr)
```

```
    if have_optical:
        ps_x            = data_mask['SEG_X']
        ps_y            = data_mask['SEG_Y']
        ps_ba           = data_mask['SEG_BA']
        ps_pa           = data_mask['SEG_PA']
        ps_radius25     = data_mask['RADIUS_R_ISO25']
        ps_radius50     = data_mask['RADIUS_R_50']
```

`photometry_wallaby.py`

Fit 2D Gaussian to the moment 9 map.

```
def gaussian_fit(input_dir, galaxy):
    '''
    input_dir:  WALLABY data products directory
    galaxy:     Galaxy name

    Returns: Gaussian RA, Dec, major/minor axes, position angle
    '''
```

Fit annuli/apertures to the moment 0 map defined by the 2D Gaussian.

```
def measure_hi_radial_profile(input_dir, galaxy, aperture_params, fignum):
    '''
    input_dir:          WALLABY data products directory
    galaxy:             Galaxy name
    aperture_params:    Parameters for annuli/apertures
                        aperture_params[0]: RA
                        aperture_params[1]: Dec
                        aperture_params[2]: Gaussian major axis
                        aperture_params[3]: Gaussian minor axis
                        aperture_params[4]: Gaussian position angle
                        aperture_params[5]: SoFiA RMS
                        aperture_params[6]: WALLABY flux correction value
    fignum:             Figure number

    Returns:  Annulus radius, area, flux, flux error, total surface density, average surface density
              Aperture radius, area, flux, flux error, total surface density, average surface density
    '''
```

Derive HI sizes and surface densities using the measured annulus/aperture fluxes. The below code is for the flux corrected values.

```
def derive_hi_size(aperture_array, annulus_array, gaussian_array):
    '''
    aperture_array:  Measured aperture values
                     aperture_array[0]:  Aperture radius
                     aperture_array[1]:  Aperture total surface density
                     aperture_array[2]:  Aperture average surface density
                     aperture_array[3]:  Aperture flux
    annulus_array:   Measured annulus values
                     annulus_array[0]:  Annulus radius
                     annulus_array[1]:  Annulus total surface density
                     annulus_array[2]:  Annulus average surface density
                     annulus_array[3]:  Annulus flux
    gaussian_array:  Gaussian parameters
                     gaussian_array[0]:  Gaussian major axis
```

```
                    gaussian_array[1]: Gaussian minor axis

    Returns: Isodensity radius, surface density, diameter, radius error
             Effective radius, surface density,
             isodensity radius (uncorrected for beam), effective radius (uncorrected for beam)
    '''
```

Save the derived quantities to a FITS table where the new columns are added after the standard SoFiA columns.

```
  if do_save_table:
    table_str   = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
    os.system('rm -rf %s' % table_str)

    tdata = []
    tcols = []
    for i in range(len(data_mask.columns.names)):
      tdata.append(data_mask[data_mask.columns.names[i]])
      tcols.append(data_mask.columns.names[i])

    tdata_1 = [galaxies, radius_iso, surfden_iso, radius_iso_err,
               radius_eff, surfden_eff,
               diameter_iso, diameter_minor, hi_ba, flux_integrate,
               radius_iso_scale, surfden_iso_scale, radius_iso_err_scale,
               radius_eff_scale, surfden_eff_scale,
               diameter_iso_scale, flux_integrate_scale,
               gaus_ra, gaus_dec, gaus_maj, gaus_min, gaus_pa]

    tcols_1 = ('OBJECTS', 'RADIUS_ISO', 'SURFACE_DENSITY_ISO', 'RADIUS_ISO_ERR',
               'RADIUS_EFF', 'SURFACE_DENSITY_EFF',
               'DIAMETER_MAJOR', 'DIAMETER_MINOR', 'AXIS_RATIO_BA', 'TOTAL_FLUX',
               'RADIUS_ISO_SCALE', 'SURFACE_DENSITY_ISO_SCALE', 'RADIUS_ISO_ERR_SCALE',
               'RADIUS_EFF_SCALE', 'SURFACE_DENSITY_EFF_SCALE',
               'DIAMETER_MAJOR_SCALE', 'TOTAL_FLUX_SCALE',
               'GAUSSIAN_RA', 'GAUSSIAN_DEC', 'GAUSSIAN_MAJ', 'GAUSSIAN_MIN', 'GAUSSIAN_PA')

    for i in range(len(tdata_1)):
      tdata.append(tdata_1[i])
      tcols.append(tcols_1[i])

    save_table_function(table_str, tdata, tcols)
```

After measuring the PanSTARRS photometry and measuring optical sizes, the H I mass and surface density within the optical disc can be measured.

```
def derive_hi_optdisc(input_dir, ps_dir, galaxy, aperture_params):
    '''
    input_dir:        WALLABY data products directory
    ps_dir:           PanSTARRS directory
    galaxy:           Galaxy name
    aperture_params:  Parameters for annuli/apertures
                      aperture_params[0]: PanSTARRS segment x position [pixel]
                      aperture_params[1]: PanSTARRS segment y position [pixel]
                      aperture_params[2]: PanSTARRS radius [arcsec]
                      aperture_params[3]: PanSTARRS segment axis ratio b/a
                      aperture_params[4]: PanSTARRS segment position angle [degrees]
```

```
    Returns:  Aperture flux, average surface density
    '''
```

A.5 Summary Plots

`galaxy_summary_plots.py`

```
def wallaby_galaxy_summary_plot(fig_num, subfig1, subfig2, subfig3, image1, image2, colour, txt_array)
    '''
    fig_num:    Figure number
    subfig1:    Number of rows
    subfig2:    Number of columns
    subfig3:    Subfigure (panel) number -- starts from 1
    image1:     Primary/background image (FITS file name - string) OR
                Integrated spectrum array image1[0]: velocity
                                          image1[1]: flux
                                          image1[2]: error
    image2:     Secondary/overlay image used to generate contours (FITS file name)
    colour:     Integrated spectrum line colour
    txt_array:  Array of parameters/properties to print or use for plotting ellipses
                txt_array[0]: galaxy name,
                txt_array[1]: sofia ID
                txt_array[2]: systemic velocity
                txt_array[3]: SoFiA RA [degrees]
                txt_array[4]: SoFiA Dec [degrees]
                txt_array[5]: log(HI mass)
                txt_array[6]: HI major axis (diameter [arcsec]
                txt_array[7]: HI minor axis [arcsec]
                txt_array[8]: 2D Gaussian fit position angle [radians]
                txt_array[9]: 2D Gaussian fit RA [degrees]
                txt_array[10]: 2D Gaussian fit Dec [degrees]
    '''
```

A.6 Photometry

A.6.1 PanSTARRS Photometry

Required input tables for measuring PanSTARRS photometry.

```
# ================================== #
# ===== Open/Join FITS Tables ===== #
# ================================== #
if ~have_segments:
  print('============ %s ============' % team_release)
   fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release
   fits_flags      = sofia_dir + '%s_catalogue_flags.fits' % team_release
  fits_gaussian   = parameter_dir + '%s_hi_structural_parameters_cm.fits' % team_release

if have_segments:
   fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release
   fits_flags      = sofia_dir + '%s_catalogue_flags.fits' % team_release
  fits_gaussian   = parameter_dir + '%s_hi_structural_parameters_cm.fits' % team_release
  fits_segments   = parameter_dir + '%s_panstarrs_segments_deblend.fits' % team_release
```

photometry_panstarrs.py

Create a segmentation map from the PanSTARRS r-band image. Target segment parameters are then used to define apertures/annuli for measuring photometry.

```
def make_segmentation_map_advanced(fits_dir, plots_dir, galaxy, band, hi_position, do_deblend):
    '''
    fits_dir:      PanSTARRS galaxy directory
    plots_dir:     Plotting directory
    galaxy:        Galaxy name
    band:          PanSTARRS image band (g or r)
    hi_position:   SoFiA HI position
                   hi_position[0]: RA
                   hi_position[1]: Dec
    do_deblend:    Flag for if the segmentation map should be deblended
                   0 - do NOT deblend
                   1 - do deblending

    Returns: Segment x, y, radius, axis ratio (b/a), position angle
    '''
```

For galaxy images requiring manual intervention to correctly find the target galaxy segment.

```
def make_segmentation_bespoke(fits_dir, plots_dir, galaxy, band, hi_position, do_seg_par, do_seg_id,
    do_deblend):
    '''
    fits_dir:      PanSTARRS galaxy directory
    plots_dir:     Plotting directory
    galaxy:        Galaxy name
    band:          PanSTARRS image band (g or r)
    hi_position:   SoFiA HI position
                   hi_position[0]: RA
                   hi_position[1]: Dec
    do_seg_par:    Specify parameters for creating segmentation map
                   do_seg_par[0]: Flag to set specific values (0/do NOT or 1/do)
                   do_seg_par[1]: nsigma to define the detection threshold
                   do_seg_par[2]: Minimum number of pixels for a segment
                   do_seg_par[3]: Source finding threshold relative to the background
                   do_seg_par[4]: Minimum segment radius
    do_seg_id:     Specify segment ID
                   do_seg_id[0]: Flag to set segment ID (0/do NOT or 1/do)
                   do_seg_id[1]: Segment ID
    do_deblend:    Specify deblending parameters (0/do NOT or 1/do)
                   do_deblend[0]: Flag to set deblending values (0/do NOT or 1/do)
                   do_deblend[1]: Contrast value
                   do_deblend[2]: Minimum segment radius (overides do_seg_par[4])

    Returns: Segment x, y, radius, axis ratio (b/a), position angle
    '''
```

Measure annulus/aperture photometry.

```
def extract_surface_brightness(survey_dir, galaxy, aperture_params, band, radius_max, fignum):
    '''
    survey_dir:       PanSTARRS galaxy directory
    galaxy:           Galaxy name
    aperture_params:  Parameters for annuli/apertures
                      aperture_params[0]: PanSTARRS segment x position [pixel]
```

```
                    aperture_params[1]: PanSTARRS segment y position [pixel]
                    aperture_params[2]: PanSTARRS segment radius [arcsec]
                    aperture_params[3]: PanSTARRS segment axis ratio b/a
                    aperture_params[4]: PanSTARRS segment position angle [degrees]
    band:           PanSTARRS image band (g or r)
    radius_max:     Maximum radius to define an aperture/annulus
    fignum:         Figure number

    Returns:  Annulus radius, area, flux (ADU), flux error (ADU), surface brightness,
              Aperture radius, area, flux (ADU), flux error (ADU), total magnitude (curve of growth),
              exposure time, sky magnitude
    '''
```

Measure isophotal radius and total g- and r-band magnitudes within an aperture defined by this radius.

```
def measure_mag_size(isophote_limit, exptime, aperture_adu, annulus_mag, radii):
    '''
    isophote_limit:   Surface brightness for measuring isophotal size
    exptime:          Image exposure time
                      exptime[0]: r-band exposure time
                      exptime[1]: g-band exposure time
    aperture_adu:     Total measured ADU from apertures
                      aperture_adu[0]: r-band ADU
                      aperture_adu[1]: g-band ADU
    annulus_mag:      r-band annulus magnitudes
    radii:            Annulus/aperture radii
                      radii[0]: annulus radii
                      radii[1]: aperture radii

    Returns:  r-band magnitude, g-band magnitude, isophotal radius
    '''
```

A.6.2 GALEX Photometry

Required input tables for measuring *GALEX* photometry.

```
# ================================== #
# ===== Open/Join FITS Tables ===== #
# ================================== #
if ~have_optical and ~have_segments:
  print('============ %s ============' % team_release)
   fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release
   fits_flags      = sofia_dir + '%s_catalogue_flags.fits' % team_release
   fits_gaussian   = parameter_dir + '%s_hi_structural_parameters.fits' % team_release

if have_segments:
  print('============ %s ============' % team_release)
   fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release
   fits_flags      = sofia_dir + '%s_catalogue_flags.fits' % team_release
   fits_gaussian   = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
   fits_segments   = parameter_dir + '%s_panstarrs_segments_deblend.fits' % team_release

if have_optical:
  print('============ %s ============' % team_release)
   fits_sofia      = sofia_dir + '%s_catalogue.fits' % team_release
```

```
  fits_flags       = sofia_dir  + '%s_catalogue_flags.fits' % team_release
  fits_gaussian    = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
  fits_panstarrs   = parameter_dir + '%s_panstarrs_photometry.fits' % team_release
```

`photometry_galex.py`

Fit annuli and apertures to the GALEX NUV-band image where the annuli/apertures are defined by the PanSTARRS segment. This function fits annuli/apertures both before and after measuring and subtracting the local sky background enabling the comparison of before/after sky background subtraction on the curve of growth.

```
def galex_surface_brightness_rmask(galex_dir, galaxy, aperture_params, band, rmax_pix, fignum):
    '''
    galex_dir:        GALEX galaxy directory
    galaxy:           Galaxy name
    aperture_params:  aperture_params[0]: PanSTARRS segment x position [pixel]
                      aperture_params[1]: PanSTARRS segment y position [pixel]
                      aperture_params[2]: PanSTARRS radius [arcsec]
                      aperture_params[3]: PanSTARRS segment axis ratio b/a
                      aperture_params[4]: PanSTARRS segment position angle [degrees]
    band:             GALEX band (NUV, FUV)
    rmax_pix:         The maximum radius at which to fit apertures/annuli
    fignum:           Figure number

    Returns: Annulus radius, area,
             background subtracted flux [ADU], background subtracted magnitude,
             flux [ADU], magnitude,
             Aperture radius, area,
             background subtracted flux [ADU], flux error [ADU], flux difference [ADU]
             background subtracted magnitude, magnitude difference,
             flux [ADU], flux difference [ADU], magnitude, magnitude difference
    '''
```

A.6.3 WISE Photometry

Required input tables for measuring WISE photometry.

```
# ================================= #
# ===== Open/Join FITS Tables ===== #
# ================================= #
if ~have_optical and ~have_segments:
  print('============ %s ============' % team_release)
  fits_sofia       = sofia_dir  + '%s_catalogue.fits' % team_release
  fits_flags       = sofia_dir  + '%s_catalogue_flags.fits' % team_release
  fits_gaussian    = parameter_dir + '%s_hi_structural_parameters.fits' % team_release

if have_segments:
  print('============ %s ============' % team_release)
  fits_sofia       = sofia_dir  + '%s_catalogue.fits' % team_release
  fits_flags       = sofia_dir  + '%s_catalogue_flags.fits' % team_release
  fits_gaussian    = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
  fits_segments    = parameter_dir + '%s_panstarrs_segments_deblend.fits' % team_release

if have_optical:
  print('============ %s ============' % team_release)
  fits_sofia       = sofia_dir  + '%s_catalogue.fits' % team_release
```

```
fits_flags      = sofia_dir  + '%s_catalogue_flags.fits' % team_release
fits_gaussian   = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
fits_panstarrs  = parameter_dir + '%s_panstarrs_photometry.fits' % team_release
```

`photometry_wise.py`

Measure WISE band magnitudes within elliptical apertures defined by the PanSTARRS segment and isophotal radius.

```
def wise_magnitude(wise_dir, galaxy, aperture_params, fignum):
    '''
    wise_dir:          WISE galaxy directory
    galaxy:            Galaxy name
    aperture_params:   aperture_params[0]: PanSTARRS segment x position [pixel]
                       aperture_params[1]: PanSTARRS segment y position [pixel]
                       aperture_params[2]: PanSTARRS radius [arcsec]
                       aperture_params[3]: PanSTARRS segment axis ratio b/a
                       aperture_params[4]: PanSTARRS segment position angle [degrees]
    fignum:            Figure number

    Returns: W1-band flux (ADU), flux error (ADU), magnitude, magnitude error,
             W2-band flux (ADU), flux error (ADU), magnitude, magnitude error,
             W3-band flux (ADU), flux error (ADU), magnitude, magnitude error,
             W4-band flux (ADU), flux error (ADU), magnitude, magnitude error
    '''
```

A.7 Derived Properties

Required input tables for `derived_properties.py`. Note: The dust extinction file `*_galactic_dust_extinction.fits` is created when running `derived_properties.py` with the switch `do_get_dust_extinction=True`.

```
fits_sofia      = sofia_dir  + '%s_catalogue.fits' % team_release
fits_flags      = sofia_dir  + '%s_catalogue_flags.fits' % team_release
fits_wallaby    = parameter_dir + '%s_hi_structural_parameters.fits' % team_release
fits_hiopt      = parameter_dir + '%s_hi_optical_disc.fits' % team_release
fits_panstarrs  = parameter_dir + '%s_panstarrs_photometry.fits' % team_release
fits_galex      = parameter_dir + '%s_galex_photometry.fits' % team_release
fits_wise       = parameter_dir + '%s_wise_photometry.fits' % team_release
fits_extinct    = parameter_dir + '%s_galactic_dust_extinction.fits' % team_release
```

A.7.1 Dust Extinction

Note that the import statement provide in the ASTROQUERY documentation to enable querying of the IRSA Dust Extinction Service is incorrect. The required line to import the IRSADUST module is shown below.

```
from astroquery.irsa_dust import IrsaDust

# ======== Galactic Dust Extinction ========== #
pos_hi          = SkyCoord(sofia_ra*u.deg, sofia_dec*u.deg, frame='icrs')
extinct_SAF     = np.full(len(galaxies), np.nan)
extinct_SFD     = np.full(len(galaxies), np.nan)
```

```
for i in range(len(pos_hi)):
    extinct_table      = IrsaDust.get_query_table(pos_hi[i], section='ebv')
    extinct_SAF[i]     = extinct_table['ext SandF mean'][0]
    extinct_SFD[i]     = extinct_table['ext SFD mean'][0]
    print(galaxies[i], 100*i/len(galaxies), extinct_SAF[i], extinct_SFD[i])
```

A.7.2 Physical Quantities

Derived physical quantities include stellar mass, H I mass, SFR, distance, H I gas fraction, H I quantities within the optical disc, ratio between galaxy disc size measured at different wavelengths (e.g. H I, optical, NUV).

```
# ==================================== #
# === Derive Physical Quantities == #
# ==================================== #
if do_derive_quantities:
    do_save_table_calc_par  = True
    do_save_table_sfr       = True

    # ======== PanSTARRS MSTAR ========== #
    gmag_sdss25         = 0.014 + 0.162 * (gmag_ps25 - rmag_ps25) + gmag_ps25
    rmag_sdss25         = 0.014 + 0.162 * (gmag_ps25 - rmag_ps25) + rmag_ps25

    gmag_sdss_ext       = gmag_sdss25 - 3.793 * extinct_SFD
    rmag_sdss_ext       = rmag_sdss25 - 2.751 * extinct_SFD

    mstar_sdss25        = calc_lgMstar(-0.840, 1.654, rmag_sdss_ext,
                                       (gmag_sdss_ext - rmag_sdss_ext),
                                       4.64, redshift, h=cosmo.h)

    # ======== NUV - r ========== #
    nuv_mag_ext         = nuv_mag - 8.2 * extinct_SFD
    nuvr                = nuv_mag_ext - rmag_sdss25

    # ======== Distance, HI Mass, HI Fraction ========== #
    distance            = dist_lum(redshift)
    mhi_msol            = hi_mass_jyhz(sofia_sint, redshift)
    hifrac              = mhi_msol - mstar_sdss25

    mhi_msol_corr       = hi_mass_jyhz(sint_corr, redshift)
    mhi_ps_iso_corr     = hi_mass_jyhz(sint_ps_disc_iso, redshift)
    mhi_ps_eff_corr     = hi_mass_jyhz(sint_ps_disc_eff, redshift)

    hifrac_corr         = mhi_msol_corr - mstar_sdss25
    hifrac_ps_iso_corr  = mhi_ps_iso_corr - mstar_sdss25
    hifrac_ps_eff_corr  = mhi_ps_eff_corr - mstar_sdss25

    # ======== Galaxy Disc Size Ratios ========== #
    sratio_hi_r25       = radius_iso_hi_corr / r25_asec
    sratio_nuv_r25      = radius_nuv / r25_asec
    sratio_hi_nuv       = radius_iso_hi_corr / radius_nuv

    # ========= Concentration Index ========== #
    cindex              = rad_90 / rad_50

    # ======== Calculate SFRs ========= #
```

```
wise_mags          = [w1_mag, w2_mag, w3_mag, w4_mag]
upperlimits        = [w3_uplim, w4_uplim, nuv_uplim]
sfr_parameters     = calculate_sfrs (wise_mags, nuv_mag_ext, distance, upperlimits)
```

Function for deriving NUV+MIR SFRs following Janowiecki et al. (2017) and WISE SFRs following Cluver et al. (2017).

```
def calculate_sfrs (wise_mags, nuv_mag_ext, distance, upperlimits):
    '''
    wise_mags:      WISE band magnitudes
                    wise_mags[0]: W1-band magnitude
                    wise_mags[1]: W2-band magnitude
                    wise_mags[2]: W3-band magnitude
                    wise_mags[3]: W4-band magnitude
    nuv_mag_ext:    NUV-band magnitude
    distance:       Luminosity distance
    upperlimits:    Flags indicating if NUV, W3 or W4 are upper limits
                    upperlimits[0]: W3 upper limit flag
                    upperlimits[1]: W4 upper limit flag
                    upperlimits[2]: NUV upper limit flag

    Returns: See README_derived_galaxy_sfrs.txt for details of returned quantities.
    '''
```

References

Bertin E., Mellier Y., Radovich M., Missonnier G., Didelon P., Morin B., 2002, in Bohlender D. A., Durand D., Handley T. H., eds, Astronomical Society of the Pacific Conference Series Vol. 281, Astronomical Data Analysis Software and Systems XI. p. 228

Bianchi L., Shiao B., Thilker D., 2017, ApJS, 230, 24

Cardelli J. A., Clayton G. C., Mathis J. S., 1989, ApJ, 345, 245

Catinella B., et al., 2018, MNRAS, 476, 875

Ciesla L., et al., 2014, A&A, 565, A128

Cluver M. E., Jarrett T. H., Dale D. A., Smith J. D. T., August T., Brown M. J. I., 2017, ApJ, 850, 68

Janowiecki S., Catinella B., Cortese L., Saintonge A., Brown T., Wang J., 2017, MNRAS, 466, 4795

Janowiecki S., Catinella B., Cortese L., Saintonge A., Wang J., 2020, MNRAS, 493, 1982

Jarrett T. H., et al., 2013, AJ, 145, 6

Lang D., 2014, AJ, 147, 108

Meisner A. M., Lang D., Schlegel D. J., 2017, AJ, 154, 161

Meyer M., Robotham A., Obreschkow D., Westmeier T., Duffy A. R., Staveley-Smith L., 2017, PASA, 34, 52

Muñoz-Mateos J. C., et al., 2015, ApJS, 219, 3

Schiminovich D., et al., 2007, ApJS, 173, 315

Schlafly E. F., Finkbeiner D. P., 2011, ApJ, 737, 103

Schlegel D. J., Finkbeiner D. P., Davis M., 1998, ApJ, 500, 525

Taylor E. N., et al., 2011, MNRAS, 418, 1587

Tonry J. L., et al., 2012, ApJ, 750, 99

Westmeier T., Deg N., Spekkens K., Reynolds T., Shen A., 2022, PASA

Willmer C. N. A., 2018, ApJS, 236, 47

Wyder T. K., et al., 2007, ApJS, 173, 293

Zibetti S., Charlot S., Rix H.-W., 2009, MNRAS, 400, 1181