PyMorph

Software for
Automated Galaxy Morphological Parameter Estimation

Vinu Vikram
School of Pure & Applied Physics
Mahatma Gandhi University
Kottayam, Kerala, India

Yogesh Wadadekar
National Centre for Radio Astrophysics
Pune, India

Ajit K. Kembhavi
Inter-University Centre for Astronomy and Astrophysics (IUCAA)
Pune, India 2008

Contents

1 PyMorph and Usage

PyMorph is a software pipeline which computes non-parametric and parametric morphological parameters of galaxies. PyMorph uses GALFIT (Peng et. al. 2002) for bulge disk decomposition of galaxy and SExtractor (Bertin et. al. 1996) for determining the initial values. PyMorph uses its own module to calculate Concentration index, Asymmetry, Clumpness, Gini Coefficient, second order moment of the brightest 20% pixels of galaxies(CASGM). [The important information to run the code can be found until section 1.9].

1.1 Dependencies

1. Python 2.7 or greater NOT python 3.0 or greater
2. numpy
3. matplotlib
4. pyfits
5. GALFIT
6. SExtractor
7. xpa (optional, if you want to select PSF)
8. MySQLdb (optional) if you need to use mysql database

1.2 Install PyMorph

- Get a copy from https://github.com/vvinuv/pymorph
- Edit .cshrc file and give

```
setenv PYTHONPATH /path/to/pymorph/pymorph
alias pymorph '/path/to/pymorph/pymorph.py'
```

1.3 Test PyMorph installation

- cd test_data
- pymorph

1.4 Run PyMorph

- cp config.py.example /your/data/area/where/you/want/to/run/pymorph
- cd /your/data/area/where/you/want/to/run/pymorph
- Edit config.py according to Section 1.7
- pymorph [options]

1.5 Working modes

The pipeline works in different modes. A short description of the available modes are the following and most of the time user may require only Normal and PSF selection modes.

1. Normal Mode : In normal mode, user will be able give two kind of inputs
   - a. A large field(s) of galaxies. eg. HST-ACS/WFPC2, SDSS, 2MASS fields
   - b. Galaxy(ies) in cutout image(s)
2. PSF Selection : To select the PSFs from image.
3. Repeat Mode [No may not need this]: The parameter estimation process can be failed due to several reasons. If the user feels that the fitting can be improved by adjusting the initial values or using an efficient mask, this mode can be used. Here the pipeline will run again on the failed galaxies using the user specified parameters and images.
4. Find and Fit [You may not need this]: Generate parameters of objects which satisfy the magnitude range specified by the user.

1.6 Basic pre-pipeline steps

1. Create a catalogue of galaxies for which the user wants to generate the morphological parameters. The possible columns in the catalogue can be seen the Section 1.9. Name of this catalog should be given in config.py (see next section). [Ignore this: If the FindAndFit mode is enabled, then the program will not search for this catalogue.]
2. Create PSF files. A file contains the list of PSF should be given in config.py (see next section)
3. Edit the config.py which is the configuration file for the pipeline. The parameters in the configuration file are described in the Section 1.7.
4. [Ignore this:] Run SExtractor on the frame and the resulting file contains the information of all the object in the frame. The output parameters of this catalogue MUST follow a particular order and that can be found in the Appendix. This is recommended, as the PyMorph may keep the sky value at the SExtractor value during the 2D decomposition. So running SExtractor needs care. If PyMorph does not find any SExtractor catalogue it will create one using the default parameters.

1.7 Input parameters

Change the values of parameters shown in red. You can play with other parameters later.

- **[imagefile:]** As described earlier, if the user is giving a large image frame (Normal mode a.) which include several galaxies to be fitted, then the name of the image should be given here.
- **[whtfile:]** The corresponding rms/weight map of the large frame. If the name contains the string *wht*/*rms* then the program treats it as weight/rms image. If this file is not found the program will skip this step.
- **[sex_cata:]** The SExtractor catalogue of all the objects in the frame (which includes both objects to be fitted and other objects). If the user provide one SExtractor catalogue, PyMorph uses it. Otherwise it generate using the default SExtractor parameters in the pipeline. One can use the command line option *–edit-conf* to change these parameters according to their images.
- **[clus_cata:]** The list of all the objects of interest, i.e. objects the user need to fit. The possible columns in this file are given in the Section: 1.9 and each column has a name (see Section: 1.9).

- **[datadir:]** The directory where you keep the input images
- **out_cata:** The name of the output catalogue. Program will make a catalogue of all the galaxies for which the morphological parameters are generated.
- **rootname:** Root name. You can give just a blank " to avoid this. If the program finds *rootname*, it will be appended to all the intermediate files and the name of the galaxy in the result file.
- **[outdir:]** The directory contains all the output
- **[psfselect:]** Since selecting GOOD PSFs (i.e. a list of good stellar images) from large frames and make a list of them is time consuming, we have added a small utility which will help the user to find the PSF with out spending much time. This can be achieved by using the psfselect parameter. This parameter can take either 0, 1 or 2. The meaning of these are following
  - 0 => The pipeline will continue with the user given PSFs.
  - 1 => The pipeline will run only for finding PSFs. This requires xpa and ds9 to be installed. The user will be prompted to accept or reject each stellar image in the frame. Finally, there will be several psf*fits files and psflist.list in the directory. psf*fits are the images of stellar images you selected and psflist.list is the list of psf*fits. While fitting pymorph select the nearest PSF to each object.
  - 2 => Find PSFs and run pipeline.

    It is recommended to use *psfselect* = 1 and select PSFs first. After having good PSFs, run pipeline with *psfselect* = 0.
- **[psflist:]** List of PSFs. You can give it either as a list like [**'psf1.fits, 'psf2.fits', etc**] or point to a file which contains the names of PSFs as **'@psflist.txt**'. The pipeline will select the nearest PSF to the object of interest either using the header information or using the information from its name. In the latter case, the PSF's name should have the form **psf_radec.fits**.

  ```
  Eg. If the PSF's position is (12:16:43.5, -12:03:12.0) then the
  name should be psf_1216435-1203120.fits.
  ```

  This convention is used to find the nearest PSF. The pipline will first check whether the mode is *repeat*. If *repeat* is false and if the program fails to find the configuration file, then it will try to find the coordinate information of the galaxy. If the program doesn't find the RA and DEC information of the galaxy, then it chooses the PSF one by one from the list.
- **mag_zero:** Magnitude zero point.
- **mask_reg, thresh_area, threshold:** Parameters for masking condition. It is explained in the Section 1.11.1.
- **size:** This parameter is a list of five quantities which controls the size and shape of the stamp image of the galaxy. The size parameters are in the order

  ```
  size = [resize, varsize, fracrad, square, fixsize]
  ```

  - *resize* - This will be used when the user supply a cutout of galaxy and wants to resize the image. This particular parameter is useful when we have a large number of frames from surveys like SDSS.
  - *varsize* - This parameter will be used to find the image cutout size. When it is true the size of the image will be decided from the half light radius of the galaxy.

- *fracrad* - Size of the cutout image will be *fracrad* times half light radius of the galaxy.
- *square* - This will decide the shape of the cutout. If it is true, then the cutout will be square otherwise a rectangle.
- *fixsize* - If the user wants to make an image of fixed size, this keyword will provide the size information.

- **pixelscale:**
- **H0, WM, WV:** Hubble parameter, Ω_M, Ω_Λ
- **back_extraction_radius:** The radius of the background region which will be used to calculate the background asymmetry and background clumpness.
- The following parameters decide the working mode of PyMorph

  repeat: [May not need this. Keep it False] If it is True, repeat the pipeline manually.

  [galcut:] True, if the input is cutout of galaxies.

  decompose: True, if the user wants to extract the structural parameters.

  cas: True, if user wants to find the CASGM parameters.

  findandfit: [Keept it False] True, to run PyMorph for galaxies within some magnitude range.

  crashhandler: [Keep it False] If it is True, then the PyMorph will handle the possible crashes during the pipeline process and try to fix those errors in the next run.

- **components:** The user can decide the components for fitting. By default PyMorph assumes a disk and a bulge to model the light distribution of the galaxy. The available components are bulge, disk and point.
- **fitting** This is also a list of three parameters which can be used to fix/fit center and sky.

  ```
  fitting = [bulge_center, disk_center, sky]
  ```

- **center_deviation:** Using this parameter, the user can specify the amount of deviation allowed for the center. If the fitted center deviates from the initial center more than this amount, that will be considered as a crash and refit the galaxy, provided the *crashhandler* is True.
- **[GALFIT_PATH:]** Path to galfit
- **[SEX_PATH:]** Path to sextractor
- **[PYMORPH_PATH:]** Path to pymorph
- **[galfitv:]** Version of galafit 2 or 3

1.8 config.py

Here is the configuration file for PyMorph.

```
"""Configure file for PyMorph. Authors: Vinu Vikram, Yogesh Wadadekar and Ajit Kembhavi (IUCAA) 2008, Alan Meert (UPenn)"""
###----Specify the input images and Catalogues----###
imagefile = 'j8f643-1-1_drz_sci.fits'
whtfile = 'j8f643-1-1_drz_rms.fits' #The weight image. If it contains the
                                    #string 'rms', this will treated as
#RMS_MAP and if it contains weight, then
                                    #that will be treated as WEIGHT_MAP.
#If nothing found, then by default it
#is treated as MAP_RMS
sex_cata = 'j8f643_sex.cat' #The sextractor catalogue which has
```

```
                                    #the format given in the file
clus_cata = 'cl1216-1201.cat' #catalogue of galaxies from
                                    #online catalogu service
                                    #(name ra1 ra2 ra2 dec1 dec2 dec3)

datadir = '/data2/home/ameert/sdss_sample/fit_data/'
#the directory containing input images
                                     #if commented out, then program uses
                                     # current directory

###----Specify the output names of images and catalogues----###
out_cata = 'cl1216-1201_out.cat' #catalogue of galaxies in the field
rootname = 'j8f643'

outdir = '/data2/home/ameert/sdss_sample/fits_out/dev/1/'
#the directory containing output data
                                      #if commented out, then program uses
                                      # current directory

###----Psf list----###
psfselect = 0 #0 => No psfselection
                                    #1 => Only Select psf
                                    #2 => Select psf and run pipeline
                                    #Recommended: Run with '1' and then run
                                    #pipeline
starsize = 20 #psf image size will be startsize times
                                    #the SMA given by SExtractor
#psflist = ['psf_1216382-1200443.fits', 'psf_1216408-1200251.fits', 'psf_1216424-1202057.fits','psf_1216487-1201246.fits','ps
psflist = '@psflist.list'
                                    #List of psf containg their
                                    #position information in the
                                    #header (RA_TARG, DEC_TARG).
                                    #Make psf with the names as here
                                    #and use psf_header_update.py.
                                    #It will update the header information.
mag_zero = 25.256 #magnitude zero point

###----Conditions for Masking----###
manual_mask = 0
mask_reg = 2.0
thresh_area = 0.2
threshold = 3.0 #Masking will be done for neighbours
                                    #whose semimajor*threshold overlaps with
                                    #threshold * semi-major axis of
                                    #the object and area of the neighbour
                                    #less than thresh_area * object area in
                                    #sq.pixel.
                                    #The masking will be for a circular
                                    #region of radius mask_reg*semi-major
                                    #axis of the nighbour with respect to
                                    #the center of the neightbour.

###---Size of the cut out and search conditions---###
###---size = [resize?, varsize?, fracrad, square?, fixsize]---###
size = [0, 1, 6, 1, 120] #size of the stamp image
searchrad = '0.3arc' #The search radius

###----Parameters for calculating the physical parameters of galaxy----###
pixelscale = 0.045 #Pixel scale (arcsec/pixel)
H0 = 71 #Hubble parameter
WM = 0.27 #Omega matter
WV = 0.73 #Omega Lambda

###----Parameters to be set for calculating the CASGM----###
back_extraction_radius = 15.0
#back_ini_xcntr = 32.0
#back_ini_ycntr = 22.0
angle = 180.0
```

```
###----Fitting modes----###
repeat = False #Repeat the pipeline manually
galcut = False #True if we provide cutouts
decompose = True
detail = False #Detailed fitting
galfit = True #Always keep this True as it is not functional yet!
cas = True
findandfit = 0
maglim = [22, 15] #if findandfit= 1, then maglim = [faint_mag, bright_mag]
stargal = 0.8 #Star-galaxy classification
crashhandler = 1

###---Galfit Controls---###
components = ['bulge', 'disk'] #The components to be fitted to the object
devauc = False # set to False to fit sersic bulge, set to true to fit devacouler's bulge (n = 4)

###---fixing = [bulge_center, disk_center, sky]
fitting = [1, 1, 0] # = 0, Fix params at SExtractor value

###----Set the SExtractor and GALFIT path here----###
GALFIT_PATH = '/home/vinu/software/galfit/modified/galfit'
SEX_PATH = '/home/vinu/software/sextractor-2.5.0/sex/bin/sex'
PYMORPH_PATH = '/home/vinu/serial_pipeline/trunk/pymorph'
galfitv = '3.0.2'

###----The following conditions are used to classify fit goo/bad----###
chi2sq = 1.9 #< chi2sq
Goodness = 0.60 #> Goodness
center_deviation = 3.0 #< abs(center - fitted center)
center_constrain = 2.0 #Keep center within +/- center_constrain

###----Database Informations----###
host = 'localhost'
database = 'Cluster'
table = 'cluster'
usr = 'vinu'
pword = 'cluster'
dbparams = ['Cluster:cl1216-1201', 'ObsID:1:int']
```

1.9 Possible columns in clus_cata

- **gal_id:** The identifier of the galaxy.
- **ra1, ra2, ra3:** The RA of the galaxy. ra1 is the degree part, ra2 is minute and ra3 is the second part.
- **dec1, dec2, dec3:** The DEC of the galaxy and have same syntax as RA
- **z:** The redshift of the galaxy
- **gimg:** The galaxy image if PyMorph runs in GALCUT mode (ie. input will be cutouts of galaxies).
- **wimg:** The corresponding weight image
- **cfile:** Configuration file for GALFIT (if user wants to run PyMorph manually)
- **ximg:** The x center of the galaxy
- **yimg:** The y center of the galaxy
- **bxcntr:** The x center of the background to find the CASGM parameters
- **bycntr:** The y center of the background to find the CASGM parameters

- **psf:** The PSF corresponding to the galaxy
- **flag:** This will be used when the *crashhandler* is on. See the Flags section to know more.

Example clus_cata

The clus_cata looks something like the following

```
gal_id ra1 ra2 ra3 dec1 dec2 dec3 mag z bxcntr bycntr ximg yimg cfile psf flag
EDCSNJ1216453-1201176 12 16 45.26 -12 01 17.6 20.663 0.7955 20.0 20.0 60.0 60.0
Gj8f647_EDCSNJ1216453-1201176.in psf_1216435-1203120.fits 128
```

Another look

```
gimg wimg ximg yimg bxcntr bycntr
Ij8f647_EDCSNJ1216453-1201176.fits Wj8f647_EDCSNJ1216453-1201176.fits 60.0 60.0 20.0 20.0
```

The minimal clus_cata

```
gimg
Ij8f647_EDCSNJ1216453-1201176.fits
```

Here we assumed that the image *Ij8f647_EDCSNJ1216453-1201176.fit*s contains a galaxy within 10 pixels radius from the center. In the case of cutouts, the minimal configuration which uses all the PyMorph functionalities is the following

```
gimg z
Ij8f647_EDCSNJ1216453-1201176.fits 0.79
```

1.10 Command line Options

Some command line options are also available and are explained as follows

- **–edit-conf (-e):** PyMorph uses default set of parameters to generate SExtractor catalogue. These parameters can affect the photometric output from SExtractor. This option allows the user to edit the SExtractor configuration file interactively.
- **–force (-f):** Normally PyMorph will not generate SExtractor catalogue if it find one. Using this option user can force the pipeline to generate SExtractor catalogue.
- **–with-psf:** By default, PyMorph will use the nearest PSF from the psflist during decomposition. User can alter this behavior by this parameter. So *–with-psf=0* takes the nearest PSF, *–with-psf=1* uses second nearest PSF and so on. By using *–with-psf=-1* one can select the farthest available PSF. This option becomes important when the user wants to test the results with different PSFs in the frame.
- **–help (-h):**
- **–lmag**, **–umag:** Magnitude constraints for GALFIT.
- **–ln**, **–un:** The minimum and maximum allowed values of Seŕsic index. Defaults are 0.1 and 20.0.
- **–lre**, **–ure:** Minimum and maximum allowed values of bulge scale length, re. Default 0 and 500 pixels.
- **–lrd**, **–urd:** Minimum and maximum allowed values of disk scale length, rd. Default 0 and 500 pixels.
- **–with-in:** Fitting will be done for objects which are NXPTS / 2 + *with-in* or NYPTS / 2 + *with-in* from the main object. By default it takes a value of 150. Usage: *–with-in=150*. **–with-filter:** Manually give the filter. This will go to the database. **–with-db:** The MySQL database name. **–with-area:** The area of PSFs.

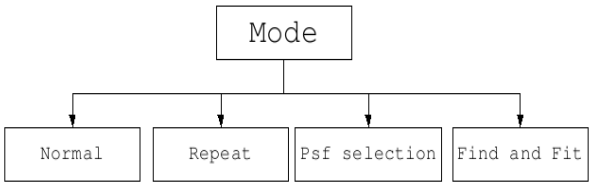


Figure 1: The PyMorph Modes

1.11 Working

The architecture of the PyMorph is shown in the Figures 1, **??** and explained as follows

1.11.1 Normal Mode with large field

PyMorph compares the user given galaxy catalogue (*clus_cata*) and SExtractor catalogue (*sex_cata*). If the pipeline finds an object in the SExtractor catalogue, it will create a stamp image and the corresponding weight map of the galaxy. Initially the pipeline try to match the RA and DEC information from the *clus_cata* with SExtractor catalogue using *serachrad*. If the *clus_cata* doesn't have RA, DEC information the pipeline will try to compare it with the pixel coordinate of the object in the frame. For that it will search for columns with headers *ximg* and *yimg*. The pipeline stops if it does not find these columns. To find the structural parameters the pipeline works as follows.

The first step in 2D bulge-disk decomposition is the creation of stamp size image of the object. The size of the image should be in such a way that the image must include enough sky and should not loose light from the outer part of the objects. An image with very large size will leads to the unnecessary usage of CPU. We use the FLUX_RADIUS (R_{50}), THETA_IMAGE (θ), ELONGATION ($\frac{a}{b}$) values from SExtractor to find the required size of the cutout. Here the FLUX_RADIUS corresponds to the half light radius (R_{50}) of the galaxy. The size of the stamp image will be found by the following formula

$$X = F_{\rm rad} R_{50} \left(|\cos\theta| + \frac{b}{a} |\sin\theta| \right) \tag{1}$$

$$Y = F_{\rm rad} R_{50} \left(|\sin\theta| + \frac{b}{a} |\cos\theta| \right) \tag{2}$$

where X and Y are the x and y dimension of the stamp image. F_{rad} is the user specified controlling parameters. In our case we found $F_{rad} = 6$ is a good compromising value for the determination of the postage stamp. We cut same portion of the noise map and use it as the weight image to GALFIT.

GALFIT allows us to fit more than one objects simultaneously. We fit a single Seŕsic function to the neighbour objects to reduce the contamination from these objects to the main galaxy. So the next step is to find the neighbour objects and determine whether it should be fitted simultaneously with the main object. To do that, we use SExtractor A_IMAGE (R) and ISO0 (A) parameters of both objects and neighbour. The criterion for simultaneous fit is the following

$$\begin{aligned} |x_o - x_n| &< T_R(R_o + R_n) \quad \text{or} \\ |y_o - y_n| &< T_R(R_o + R_n) \quad \text{or} \\ A_n &> T_A A_o \end{aligned} \tag{3}$$

where x_o, x_n, y_o, x_n are the x and y centers of object and neighbour and R_o, R_n, A_o and A_n are the A_IMAGE and ISO0 of object and neighbour. We found the controlling parameters $T_R = 3.0$

and $T_A = 0.3$ are promising. We mask all the neighbour objects which does not satisfy the above criterion (Eqn 3). We mask all the pixels of the neighbour objects which satisfy $r < T_M R_n$, where r is the radius in an elliptical aperture. We have used $T_M = 2.0$ so that all the pixels of the neighbour object will be masked. This masking technique will work only if SExtractor detect the neighbour object. Since the detection depends on the DETECT_THRESH and DETECT_MINAREA parameters, it is possible that some furious pixels can left undetected by SExtractor. So we use the following simple technique to mask those furious pixels. From the center of the main object we make elliptical annuli with increasing radii. In the inner aperture we find the maximum value of the galaxy. We assume a smooth distribution for the galaxy's light profile which decreases from the center. This implies the largest value in the central elliptical aperture is the maximum value the object can have. So we mask all the other pixels outside the inner aperture with value greater than this maximum value. Now we go to next annulus and find the maximum and mask other pixels outside this aperture which has value larger than the maximum of this aperture. This procedure continue till the aperture radius hit the image limit. Then we use a set of binary morphology operations to clean the mask. Now almost all the furious pixels and undetected objects will be masked properly and we combine this with the neighbour mask.

Next step is to find initial values for the fitting parameters. To each galaxy in the catalogue we fit a Seŕsic and exponential functions. The Seŕsic function which is used to model the luminosity profile of the bulge part is given by

$$I(r) = I_e e^{-b_n(\frac{r}{r_e}^{1/n}-1)} \tag{4}$$

and the exponential function which model the disk part of the galaxy is given by

$$I(r) = I_d e^{-\frac{r}{r_d}} \tag{5}$$

where r_e is the bulge scale length, n is the Seŕsic index and r_d is the disk scale length. b_n is a parameter which depends on the Seŕsic index.

GALFIT accepts the initial values of the total magnitude, scale radii, axis ratios and position angles of bulge and disk and Seŕsic index of the bulge. We set MAG_AUTO value from SExtractor as the input value to the total magnitudes of both bulge and disk. For the scale radii we give the half light radius of the galaxy. By default PyMorph set the initial value for Seŕsic index (n) to 4, which corresponds to de Vaucouleurs' law. After setting the initial value we create a contrain file, which restrict the free parameters from going to unphysical values during the fit. The position angle and axis ratio are set from the values of the THETA_IMAGE and ELONGATION parameters.

The estimation of CASGM parameters are explained in the Section 2. Finally the pipeline creates diagnostic plots (Figures 3, 4) and results in different formats which includes html, csv, mysql database and fits cutouts.

1.11.2 Normal Mode with cutouts

In this mode the pipeline follows the same route as explained in the previous section. In this case if the program doesn't find the RA, DEC information or the centroid of the object, it extract the morphological parameters of the object in the center of the image.

1.11.3 Repeat Mode

During this mode of run the pipeline will not create or modify the mask image or GALFIT configuration file, if they exist. So one can adjust his GALFIT configuration file / mask image before running the pipeline in the REPEAT mode.

1.11.4 Find and Fit

In this mode user can fit objects without creating *clus_cata*. The user has to give the magnitude range of the object to be fitted. This is useful when one wants to find the quantitative morphology of all the objects in the frame.

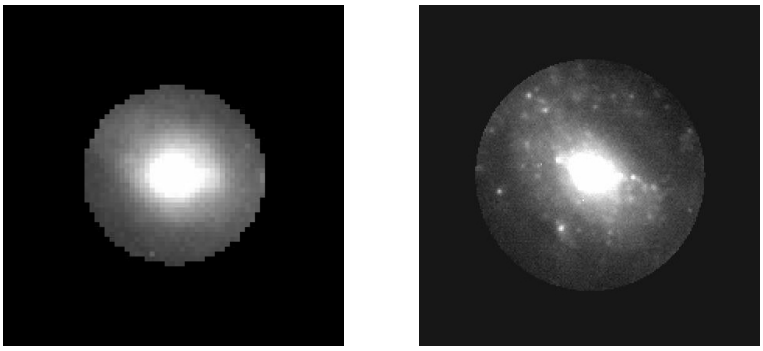
1.12 Filenames

The PyMorph makes a number of files and those filenames follow unique format. The filename convention is illustrated below. Suppose in the config.py the parameter *rootname* = j8f645 and *gal_id*, which is the name of the galaxy in the *clus_cata* is 9999, then

- **Ij8f645_9999.fits:** The cut out of the galaxy.
- **Wj8f645_9999.fits:** Corresponding weight image.
- **Mj8f645_9999.fits:** Mask image for GALFIT.
- **EMj8f645_9999.fits:** Mask image for ellipse fitting.
- **EMj8f645_9999.fits.p:l** EMj8f645_9999.fits will be converted to EMj8f645_9999.fits.pl for ellipse task.
- **Gj8f645_9999.in:** Configuration file for GALFIT.
- **Oj8f645_9999.fits:** The output image from GALFIT.
- **fit2.log:** The output parameters will be append to this file
- **error.log:** The status of the process. From this file the user gets the information about crashes.
- **E_j8f645_9999.txt:** 1D surface brightness profile of the input image.
- **OE_j8f645_9999.txt:** 1D surface brightness profile of the model.
- **P_j8f645_9999.png:** The plot of input, output, residue images and the 1-D profile comparison.
- **R_j8f645_9999.html:** The html output.
- **index.html:** The index file of all the objects.
- **result.csv:** The csv file contains all the parameters
- **agm_result_with_radius.csv:** The file contains the radial variation of Asymmetry , Gini coefficients and M20. Also r_{20}, r_{50}, r_{80}, r_{90}, Petrosian radius etc. are included.
- **restart.cat:** The catalogue contains all the failed objects with the corresponding lines in the *clus_cata*. This catalogue can be used to restart the PyMorph in the case of failed galaxies.
- **CRASH.CAT:** Probably the user may not want to use this. This will be used by the pipeline if the *crashhandler* is True.

2 CASGM Module

Concentration, Asymmetry, Clumpness, Gini coefficient and Moment of the galaxy (CASGM) are widely used to generate quantitative galaxy morphology, for the last few years. In this section we describe the implementation of these parameters in **PyMorph**.



20% light contained portion 80% light contained portion

Figure 2: Portion of galaxy within r_{20} and r_{80}

2.1 Concentration (C)

1. Find the Petrosian radius r_η at which parameter (η) becomes 0.2. The Petrosian parameter, η, is defined as

$$\eta = \frac{L(R)}{L(< R)} \tag{6}$$

L(R) is the average surface brightness at the radius R and $L(< R)$ is the average brightness inside the radius R.

2. Find the total light of the galaxy as the light inside $1.5 \times r_\eta$.
3. Find the 80% and 20% light contained radii. ie. r_{80} and r_{20}.
4. The concentration index will be found using the equation

$$C = 5 \log(\frac{r_{80}}{r_{20}}) \tag{7}$$

2.2 Asymmetry (A)

1. Define an extraction region of radius $1.5 \times r_\eta$.
2. Rotate the image [1] by 180^0
3. Find the asymmetry parameter using the equation

$$A = \frac{\sum |I_0 - I_\phi|}{2 \sum |I_\phi|} \tag{8}$$

where I_ϕ is the rotated image and I_0 is the original image.

4. Centering correction will be applied by minimizing the A value w.r.t. center.
5. Noise correction will also be done by subtracting the asymmetry of the background from the image asymmetry.

2.3 Clumpness (S)

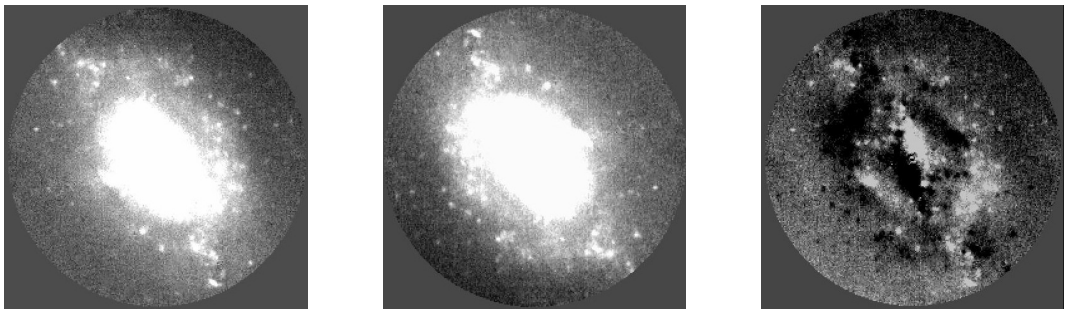
1. Smooth the image with a boxcar of size $r_\eta/4$
2. Remove the center region of the galaxy within a radius $r_\eta/4$ as the center is not always resolved.
3. Clumpness parameter can be computed using the equation

$$S = 10 \times \frac{I - I^\sigma}{I} \tag{9}$$

where I is the original image and I^σ is the smoothed image.

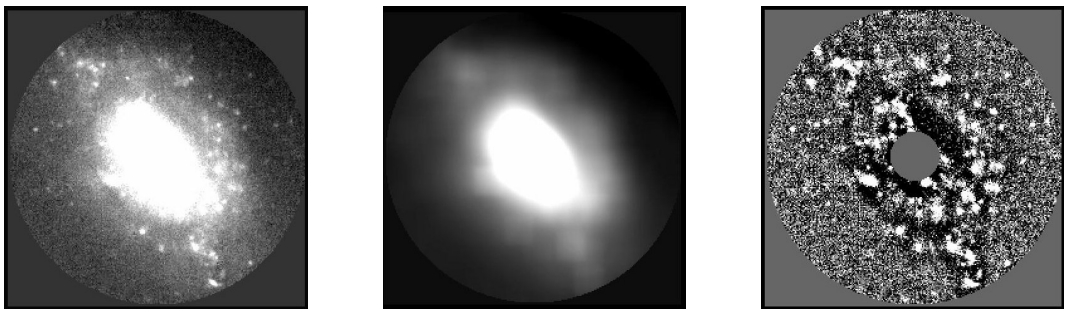
4. Subtract the background clumpness to get the final clumpness.

[1] The image used here is of NGC 5585 in R band.



Original image Rotated image Residual image

Figure 3: Images within the extraction radius



Original image Smoothed image Residual image

Figure 4: Images within the extraction radius

2.4 Gini coefficient (G)

Gini coefficient quantifies galaxy's light distribution among the pixels. Its value lie in between 0 and 1. If all the light is concentrated in one pixel, then G will be 1 and if the light distributed uniformly among the pixels, then G will be zero. To find Gini coefficient we use the following technique

1. Create a segmentation map, ie, find the pixels belong to the galaxy. To do this we smooth the image by a boxcar of size $r_\eta/5$. This will increase the signal-to-noise ratio at the outer regions of the galaxy. The surface brightness μ_η at r_η is measured and pixels in the smoothed image with flux values greater than μ_η and less than 10σ is assigned to the galaxy. σ is the sky deviation in the image. The upper limit assures that any remaining cosmic rays or spurious noise pixels in the image are not included in the segmentation map.

2. Sort the pixels in the segmentation map according to their photon counts and the Gini coefficient will be computed using the equation

$$G = \frac{1}{\overline{X}n(n-1)} \sum_{i}^{n} (2i - n - 1)X_i \tag{10}$$

2.5 The Moment of the Light (M20)

M_{20} is the second order moment of the brightest 20% pixels of the galaxy.

1. Find the second-order moment M_{tot} of the galaxy. Here only the pixels belongs to segmen-

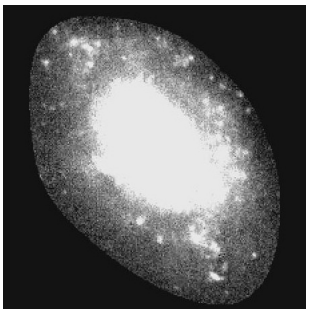


Figure 5: Segmentation map of the galaxy

tation map will be considered. We use the following equation to find M_{tot}

$$M_{tot} = \sum_{i}^{n} M_i = \sum_{i}^{n} f_i \left[(x_i - x_c)^2 + (y_i - y_c)^2 \right] \tag{11}$$

where x_c, y_c is the galaxy's center.

2. Minimize M_{tot} w.r.t. center of the galaxy.

3. Sort the pixels by flux, sum M_i over the brightest pixels until the sum of the pixel values equals 20% of the total galaxy flux, and then normalize by M_{tot}.

$$M_20 = \log \left(\frac{\sum M_i}{M_{tot}} \right) \tag{12}$$

while $\sum_i f_i < 0.2 f_{tot}$

where f_{tot} is the total flux of the segmentation map. The normalization by M_{tot} removes the dependence on total galaxy flux or size.

3 Parallel PyMorph

We have parallelized the pipeline based on Single Program, Multiple Data technique. The algorithm used here is shown in the Figure **??**. In this implementation we will have a master processor and N_P slaves. The master will do the pre-processing and post-processing. In the pre-processing part the master processor will create stamp image of the galaxy and pass it to the slave. This process continues until the number of jobs (N_J) less than N_P. The slave runs PyMorph in the GALCUT mode and send the result to the master. As soon as the master get the result from the N^{th} slave it creates cutout of another galaxy and send to the same processor. Finally, the master generate a final result file which is the only post-processing part. We have tested the parallel PyMorph and found that if we increase the number of slaves 10 fold, then the time requires to generate the parameters decreases $\sim$ 6 fold.

4 Appendix

SExtractor needs a configuration file, output parameters file, convolution kernel file and Neural Network file for Star/Galaxy classification files for its execution. PyMorph uses the following files as default.

4.1 SExtractor Configuration File

```
#------------------------------- Catalog ------------------------------------

CATALOG_NAME     j8f631_sex.cat       # name of the output catalog
CATALOG_TYPE     ASCII_HEAD     # NONE,ASCII,ASCII_HEAD, ASCII_SKYCAT,
```

```
                              # ASCII_VOTABLE, FITS_1.0 or FITS_LDAC
PARAMETERS_NAME  default.param  # name of the file containing catalog contents

#------------------------------- Extraction ----------------------------------

DETECT_TYPE      CCD            # CCD (linear) or PHOTO (with gamma correction)
DETECT_MINAREA   6              # minimum number of pixels above threshold
DETECT_THRESH    1.5            # <sigmas> or <threshold>,<ZP> in mag.arcsec-2
ANALYSIS_THRESH  1.5            # <sigmas> or <threshold>,<ZP> in mag.arcsec-2

FILTER           Y              # apply filter for detection (Y or N)?
FILTER_NAME      default.conv   # name of the file containing the filter

DEBLEND_NTHRESH  32             # Number of deblending sub-thresholds
DEBLEND_MINCONT  0.005          # Minimum contrast parameter for deblending

CLEAN            Y              # Clean spurious detections? (Y or N)?
CLEAN_PARAM      1.0            # Cleaning efficiency

MASK_TYPE        CORRECT        # type of detection MASKing: can be one of
                                # NONE, BLANK or CORRECT

#------------------------------- Photometry ----------------------------------

PHOT_APERTURES   5              # MAG_APER aperture diameter(s) in pixels
PHOT_AUTOPARAMS  2.5, 3.5       # MAG_AUTO parameters: <Kron_fact>,<min_radius>
PHOT_PETROPARAMS 2.0, 3.5       # MAG_PETRO parameters: <Petrosian_fact>,
                                # <min_radius>
PHOT_FLUXFRAC    0.5            # flux fraction[s] used for FLUX_RADIUS
SATUR_LEVEL      100000.0        # level (in ADUs) at which arises saturation
MAG_ZEROPOINT    25.256           # magnitude zero-point
MAG_GAMMA        4.0            # gamma of emulsion (for photographic scans)
GAIN             1.0            # detector gain in e-/ADU
PIXEL_SCALE      0            # size of pixel in arcsec (0=use FITS WCS info)

#------------------------- Star/Galaxy Separation ----------------------------

SEEING_FWHM      0.11            # stellar FWHM in arcsec
STARNNW_NAME     default.nnw    # Neural-Network_Weight table filename

#------------------------------- Background ----------------------------------

BACK_SIZE        64             # Background mesh: <size> or <width>,<height>
BACK_FILTERSIZE  3              # Background filter: <size> or <width>,<height>

BACKPHOTO_TYPE   GLOBAL         # can be GLOBAL or LOCAL

#------------------- Memory (change with caution!) -------------------------

MEMORY_OBJSTACK  3000           # number of objects in stack
MEMORY_PIXSTACK  300000         # number of pixels in stack
MEMORY_BUFSIZE   1024           # number of lines in buffer

#----------------------------- Miscellaneous ---------------------------------

VERBOSE_TYPE     NORMAL         # can be QUIET, NORMAL or FULL
WRITE_XML        N              # Write XML file (Y/N)?
XML_NAME         sex.xml        # Filename for XML output

#------------------------------- Check Image ----------------------------------

CHECKIMAGE_TYPE  APERTURES           # can be NONE, BACKGROUND, BACKGROUND_RMS,
                                # MINIBACKGROUND, MINIBACK_RMS, -BACKGROUND,
                                # FILTERED, OBJECTS, -OBJECTS, SEGMENTATION,
                                # or APERTURES
CHECKIMAGE_NAME  check.fits     # Filename for the check-image

#------------------------------- WEIGHTing ----------------------------------
```

```
WEIGHT_TYPE      MAP_RMS           # type of WEIGHTing: NONE, BACKGROUND,
                                   # MAP_RMS, MAP_VAR or MAP_WEIGHT
WEIGHT_IMAGE     j8f631_drz_rms.fits    # weight-map filename
WEIGHT_GAIN      N                 # modulate gain (E/ADU) with weights? (Y/N)
```

4.2 SExtractor Output Parameters

```
NUMBER
X_IMAGE
Y_IMAGE
ALPHA_SKY
DELTA_SKY
FLUX_ISO
FLUXERR_ISO
MAG_ISO
MAGERR_ISO
FLUX_RADIUS
BACKGROUND
THETA_IMAGE
ELONGATION
ISOO
A_IMAGE
FLAGS
CLASS_STAR
MAG_BEST           Uses in the case of findandfit mode
```

4.3 SExtractor Convolution Kernel

```
By default PyMorph uses 5x5 convolution mask of a Gaussian PSF with FWHM = 2.5 pixels.
```

4.4 SExtractor Neural Netwrok File

```
PyMorph uses the default.nnw file coming with SExtractor
```

4.5 PyMorph Flags

The flags used in PyMorph are the following

Flag	Explanation
1	Repeat Mode
2	Fit bulge center
4	Fit disk center
8	Fit sky
16	The cutimage extend goes outside the image
32	Galaxy ellipse failed
64	CASGM module failed
128	Galfit failed
256	Plotting failed
512	Fitting bulge
1024	Fitting disk
2048	Fitting point
4096	Neighbour fit
8192	Large chisq
16384	Low goodness
32768	Fake center
65536	Seŕsic parameter hit the limit
131072	Disk parameter hit the limit
262144	Asymmetry is not Converged
524288	Asymmetry calculation goes outside frame
1048576	Background region determination is poor