PyMorph

Table of Contents

PyMorph

[Python MORphological Parameters Hunter]

Authors: Vinu Vikram, Yogesh Wadadekar, Ajit K. Kembhavi

Softwares Required

```
1. Python 2.4 or greater
2. Stsci_python (This package includes numpy, pyfits, pyraf)
3. GALFIT
4. SExtractor
5. xpa (optional, if you want to select psf)
```

Working modes

The pipeline will work in different modes. They are describing as follows.

1. **Normal Mode**

```
a. Galaxy(ies) in a large field
b. Galaxy in a cutout image
```

2. **Repeat Mode**

```
The fitting process can be failed due to several reasons. If we feel the fitting
can be improved by adjusting the initial values or using an efficient mask, this
 mode can be used.
```

3. **Find and Fit**

```
Fit objects which in some magnitude range.
```

4. **Psf Selection**

```
a. Run PyMorph to find and extract psf from the image
b. Find psf and run PyMorph
```

Pre-Pipeline Procedure

1. Run SExtractor on the frame and the resulting file contains the information of all the object in the frame. The output parameters of this MUST follow a particular order and that can be found in the appendix. This process is recommended as the PyMorph may keep the sky value at the SExtractor value during the decomposition. So running SExtractor needs care. In case if the PyMorph does not find any SExtractor catalogue it will make one using the default parameters.
2. Make a file which contains the position, redshift information etc. of the galaxies which we are going to fit.
3. Make psf. Either using PyMorph or by some other means
4. Edit the config.py which the configuration file for the pipeline. The parameters in the configuration file are described below

Input parameters

- **imagefile:**
  - The frame contains the galaxies. This will use only if you are decomposing galaxies in a large frame.
- **whtfile:**
  - The corresponding rms weight map of the large frame. If this file is not found the program will skip this step.
- **sex_cata:**
  - The SExtractor catalogue of all the objects in the frame. In the case of large field you MUST supply the sextractror catalogue.
- **clus_cata:**
  - The list of all the obects of interest. The possible columns in this file is here and each column should have the title. The program need atleast gal_id or gimg to run.
- **out_cata:**
  - The name of the output catalogue. This file is used to write all the galaxies detected by the program during the run.
- **rootname**
  - Root name. You can give just a blank '' to avoid using this.
- **psfselect**
  - Since selecting GOOD psf and make a list of them is difficult, we have added a small utility which will help the users to find the psf with out spending much time. This can be achived by using the psfselect parameter. This parameter can take either 0, 1 or 2. The three possibilities are as follows
    - 0 => No psf selection, ie. the pipeline will continue with the user supplied psfs.
    - 1 => Only Select psf. The pipeline will run only for selecting psfs.
    - 2 => Select psf and run pipeline.
  - It is recommended to use psfselect = 1 and select psf. After having good psf, continue pipeline run using psfselect = 0. If you are hurry use psfselect = 2
- **starsize**
  - The size of the psf image in terms of the semi-major axis of the image. The size of the image will be *starsize * semi-major axis*
- **psflist:**
  - List of psfs. You can give it as a list like **['psf1.fts, 'psf2.fits', etc]** or give a file contains the psf name as **'@psflist.txt'**. The pipeline will select the nearest psf to the fitting galaxy either using the header or using the information from its name. If in the latter case, the psf's name should be in 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:**

- Masking will start for neighbors whose distance from the object greater than threshold * semi-major axis of the object and area of the neighbor less than thresh_area sq.pixel. The masking will be done for a circular region of radius mask_reg * semi-major axis of the neighbor with respect to the center of the neighbour. In the case of large frame, it is possible that some light from objects from outside the cutout can also contaminate the cutout. In that case the program is intelligent enough to mask those region and elliptical masking will be used for those cases.

- **size:**
    - This parameter is a list of five parameters 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 and wishes to resize that image. This particular parameter is useful when we have a large number of individual galaxy images from surveys like SDSS.
    - *varsize* - This parameter will be used to find the right image cutout size. When it is true the size of the image will be decided by using the half light radius.
    - *fracrad* - The size of the image w.r.t. the half light radius. Size of the image will be *fracrad* times half light radius of the galaxy.
    - *square* - This will decide whether the cutout is rectangular shaped or square shaped. For square shape, this will be 1.
    - *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, Omega matter, Omega lambda
- The following parameters are used for calculating the CASGM parameters.
    - **back_extraction_radius:**
        - The radius of the background region
    - **angle:**
        - The angle of rotation for the calculation of asymmetry
- The following parameters determine which mode should be used for pipeline
    - **repeat:**
        - Repeat the pipeline manually, if it is True
    - **galcut:**
        - True if we provide cutouts
    - **decompose:**
        - True, if you need 2D bulge disk decomposition
    - **cas:**
        - True, if you need casgm parameters
    - **findandfit**
        - '1', to use this mode otherwise '0'
    - **crashhandler**
        - If it '1', then the PyMorph will handle the possible crashes and try to fix. The details can be found in the section **Working**
- **components:** The user can decide the components for fitting. By default PyMorph will with a disk and a bulge to the object. The available componets are bulge, disk and point.
- **fitting** This is also a list of three parameters which can be used to fix/fit center and sky.

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

The parameter are self explanatory.

- The following parameters are used to classify good/bad fit.
    - chi2sq:
        - Good fit if the Chi2Nu < chi2sq
    - Goodness :
        - Good fit if the Goodness > Goodness
    - center_deviation:
        - Good fit if abs(center - fitted center) < abs(center - fitted center)

config.py

```
"""Configure file for PyMorph. Authors: Vinu Vikram, Yogesh Wadadekar and Ajit Kembhavi 2008"""
###----Specify the input images and Catalogues----###
imagefile = 'j8f643-1-1_drz_sci.fits'
whtfile = 'j8f643-1-1_drz_rms.fits'   #The weight image.
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)

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

###----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']
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
galfit = True #Always keep this True as it is not functional yet!
cas = True
findandfit = 0
crashhandler = 0

###---Galfit Controls---###
components = ['bulge', 'disk']        #The components to be fitted to the objec
###---fixing = [bulge_center, disk_center, sky]
fitting = [0, 0, 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'

###----The following conditions are used to classify fit goo/bad----###
chi2sq = 1.9                          #< chi2sq
Goodness = 0.60                       #> Goodness
center_deviation = 5.0                #< abs(center - fitted center)
```

The parameters in clus_cata

1. **gal_id:**
    - The identifier of the galaxy.
2. **ra1, ra2, ra3:**
    - The RA of the galaxy. ra1 is the degree part, ra2 is minute and ra3 is the second part.
3. **dec1, dec2, dec3:**
    - The DEC of the galaxy and have same syntax as RAM
4. **z:**

- The redshift of the galaxy

5. **gimg:**

   - The galaxy image

6. **wimg:**

   - The corresponding weight image

7. **cfile:**

   - Configuration file for GALFIT

8. **ximg:**

   - The x center of the galaxy

9. **yimg:**

   - The y center of the galaxy

10. **bxcntr:**

    - The x center of the background for finding the CASGM parameters

11. **bycntr:**

    - The y center of the background for finding the CASGM parameters

12. **psf**

    - The psf corresponding to the galaxy

13. **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 have assumed that the image Ij8f647_EDCSNJ1216453-1201176.fits contains a galaxy within 10 pixels radius from the center.

In the case of cut outs The minimal configuration which uses all the pipeline utilities is the following

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

Command line Options

Some command line options are also available.

--edit-conf (-e) PyMorph use some default set of parameters to generate SExtractor catalogue. Since these input parameters affect the SEXtractor output and so the fit, the users are asked to make there own SExtractor catalogue. 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 generate SExtractor catalogue always.

--with-psf (-p) 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 soon. Using --with-psf=-1 user can use the farthest available psf. This will become particulary important in the case of testing psf variation over a large field / consistency of decomposition with psf.

--help (-h) Help on running pymorph with option

--lmag, **--umag** Minimum and maximum magnitudes allowed during fitting. By default lmag = 100 and umag = -100. Same range will be used for both bulge and disk.

--ln, **--un** The minimum and maximum allowed values of Sersic index.

--lre, **--ure** Minimum and maximum allowed values of bulge scale length, re.

--lrd, **--urd** Minimum and maximum allowed values of disk scale length, rd.

Working

The architecture of the PyMorph is show in this figures and explained as follows

Normal Mode with large field

- It compare the galaxy catalogue (clus_cata) and sextractor catalogue (sex_cata) and if the pipeline find an object in the sextractor catalogue, it will make a stamp image and the correspoding weight map of the galaxy. The pipline first try to match the RA and DEC information in clus_cata with sextractor catalogue. If the clus_cata doesn't have any of the ra1, ra2, ra3 and dec1, dec2, dec3 column, the pipeline will try to compare it with the physical coordinate of the object in the frame. So it will search for columns with headers ximg and yimg. If these colums are also unavailable the pipeline will not find any objects in the case of large frame and exit.

- The pipeline will find the neighbour objects of the galaxy from the SExtractor catalogue.
- It makes a mask using the parameters supplied in the configuration file.
- It makes configuation file for running GALFIT using the SExtractor catalogue. Here the object will be fitted with Sersic + Exponential function and neighbours will be fitted by a single Sersic function.
- Run GALFIT
- Find the Physical parameters from the fitted one.
- It makes a mask for Ellipse task. The mask for Ellipse task and that of GALFIT are different. In the case of Ellipse mask all objects near the galaxy will be masked and the pipeline will use the ellipticity and position angle information to do that. But in the case of the GALFIT mask a circular masking will be done according to the parameters supplied in the configuration file.
- Run Ellipse task on the galaxy image using the SExtractor parameters as the initial values.
- Run Ellipse task on the model image of the galaxy.
- Compare the two 1-D profiles.
- It makes plots of galaxy, model, residual, histogram, mask and the 1-D profiles.
- It makes an html file and csv file contains the fitted parameters and casgm parameters.

Normal Mode with cutouts

- In this case also the pipeline does all the works as explained in the previous section. In addition to those, if you are supplying cutouts of galaxies, then the pipeline assumes the center of the object lies in the center of the cutout and assign the values of ximg and yimg as size/2.

Repeat Mode

- In this mode, the pipeline assumes there is cutout of galaxies as it has made during the previous run. So if the clus_cata contains the colums gimg or gal_id, the pipeline runs for that galaxies. During this mode the pipeline will not make/alter any mask image and galfit configuration file, if they exists. So you can adjust your GALFIT configuration file / masking before running the pipeline in the REPEAT mode

Find and Fit

- In this mode user can fit objects without creating clus_cata. PyMorph will ask the user some necessary information like the magnitude range, redshift and the object classification probability and find the morphological parameters. The user must create psf before going to run in this mode.

Psf Selection

- One of most difficult problem during the Morphological parameter estimation is to get good psf. Even in the case of PyMorph the situation won't differ much. But PyMorph is providing a very handy tool to select the psf out of the frame. As one the collaborator tells, this procedure is something like playing computer game. It is interesting but need much care. The keywords in config.py, 'psfselect' and 'starsize' are the controlling parameters of the mode. By default PyMorph will find the nearest psf from the psf list. This will cause some problem while you are using cut image, where you will have one psf corresponding to one galaxy. Taking this in to account PyMorph will update the clus_cata with one psf to each galaxy under the column 'psf'.

Crash Handler

If the parameter crashhandler is on in the config.py, it will be invoked in three situations

- Galfit crashes or one of the bulge / disk parameter hits the limit
    - Solution: Try to fit again with the following conditions
        - Fix / free sky, if it is free / fix
        - Fix / free centers of bulge and disk, if the centers are found free / fix
- Reduced chi square is large
    - Solution: Fix / free centers of bulge and disk, if the centers are found free / fix
- Fake center
    - Solution: Fix centers of bulge and disk.

Filenames

The PyMorph will output a number of files and those filenames has adopted a 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**
    - Correspoding weight image for the cuts.
- **Mj8f645_9999.fits**
    - Galfit mask.
- **EMj8f645_9999.fits**
    - Mask for ellipse task.
- **EMj8f645_9999.fits.pl**
    - 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 ouput image from galfit.
- **fit2.log**
    - The output parametrs will be append to this file
- **error.log**
    - The process status of the pipeline can be seen in the file
- **E_j8f645_9999.txt**
    - The ellipse task output of input image.
- **OE_j8f645_9999.txt**
    - The ellipse task output of output image.
- **P_j8f645_9999.png**
    - The plot of input, output, residue images and the 1-D profile comparison.
- **R_j8f645_9999.html**
    - The html output including the figures and parameters.
- **index.html**
    - The index file of all the fit will be in this.
- **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
- **restart.cat**
    - The catalogue contains all the 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 in the case of crash handling.

Masking Method

In PyMorph masking will be done separately for ellipse task and for decomposition. In the case of ellipse task all the neighbors are masked using the SExtractor information. But SExtractor can be failed to resolve small objects near the brighter ones. In that case the PyMorph will try to find those using the following method.

- It will find the maximum value inside a small radius of the object of our interest.
- It will search any other pixels out side the small radius above the maximum. If there are something it will mask mask those pixels. Then using that mask, the image will be masked. The central part where the maximum is found will also be masked. Then the radius will be increased further and again find the maximum inside that. This will continue till the image boundary.
- If it doesn't find any pixels above the maximum, the program will increase the radius and go on.
- After it reaches the image boundary, using the **ndimage** *fill_hole* and *erosion* functions suitable operations on masking will be done. This will remove one pixel mask etc. In the case of masking for decomposition, only object which doens't fit will be masked.

CASGM Parameters

CASGM is the short form of Concentration, Asymmetry, Clumpness, Gini coefficient and Second order Moment of the galaxy. The algorithm for these parameters are explained below.

Concentration

1. It calculates the average light at different radii. i.e., the average light in an annular ring at different radii.

2. It calculates the average light inside the apertures of different radii.

3. From the above two it calculates the petrosian eta(r) value. Petrosian ratio at a radius r from the center of an object to be the ratio of the local surface brightness in an annulus at r to the mean surface brightness within r

    eta(r) = I(r) / <I(r)>

4. Find the radius at which the Petrosian equal to 0.2

5. Compute the light inside the aperture of radius 1.5 times the Petrosian radius, that contains more than 90% of the galaxy's total light.

6. Find the 20%, 50% and 80% light radii. Linear interpolation is used for this.

7. Compute concentration parameter as 5*log(r(80%)/r(20%))

Asymmetry

1. Rotate the galaxy through 180 degrees about its center. Bilinear interpolation was used to find out the rotated image.

2. Extract a circular region of the image of size 1.5 times the Petrosian radius of the galaxy.

3. Find the residue of the two images and find the asymmetry value

    A = Sum(abs(I_0 - I_r) / Sum(I_0)

    where I_0 is the galaxy pixel value and I_r is that of rotated image

4. Centering correction:
    1. Asymmetry is computed for centers at the surrounding eight points in a 3X3 grid
    2. This procedure repeats until a minimum is found for the asymmetry.

5. Noise correction:
    1. The uncorrelated noise can be corrected by substracting the asymmetry of the background.

6. The final formula to compute asymmetry is

    ```
    A = min(Sum(abs(I_0 - I_r) / Sum(abs(I_0)) - min(Sum(abs(B_0 - B_r) / Sum(abs(I_0))
    ```

    where B_0 is the background pixel value and I_r is that of rotated background

Clumpness

1. The image is smoothed by a boxcar of width

    0.25 * r(Petrosian parameter = 0.2)

2. The smoothness is computed with the radius 1.5 by using

    S = 10 * Sum(I_0 - I_S) / Sum(I_0)

    where I_0 is the galaxy pixels and I_S that of smoothed image

3. Compute the average smoothness of the background and subtract from S.

4. The inner region of the galaxy is not considered in the computation of S as these are often unresolved.

5. Use only the positive pixels for the computation.

Gini Coefficient

1. Find the pixels in the image which belong to the galaxy, ie. make a segmentation map. This can be done by smoothing the image by a boxcar of size r(\eta)/5

2. The surface brightness at r(\eta), I_(\eta) is measured and pixels in the smoothed image with flux values greater than I_(\eta) and less than 10(\sigma) is assigned to the galaxy. \sigma is the sky deviation and which removes any remaining cosmic rays or spurious noise pixels in the image and are not included in the segmentation map.

3. The Gini coefficient can be computed by the equation

```
G = (1 / Avg(X) * n * (n-1)) * Sum over pixel[(2 * i - n -1 ) * X]
```

M20

1. The total second-order moment M_tot is the flux in each pixel f_i multiplied by the squared distance to the center of the galaxy, summed over all the galaxy pixels assigned by the segmentation map.

```
M_tot = Sum(f_i * [(x_i - x_c)^2 + (y_i - y_c)^2])
```

Where xc, yc is the galaxy's center.

2. The center is computed by finding xc, yc such that M_tot is minimized.

3. Define M20 as the brightest 20% of the galaxy's flux.

4. To compute M20, sort the pixels by flux, sum M_i over the brightest pixels until the sum of the brightest pixels equals 20% of the total galaxy flux, and then normalize by M_tot.

Sky Sigma and Background region

If the user supply the background center, PyMorph will find the sky deviation from that region. But if these parameters are not given, then the Pymorph will calculate the sky deviation first and then using this find the background region. To find the sky deviation, the PyMorph will first mask all the object detected in the cutout. Then using that mask find the sky deviation. Since the estimation of CASGM parameters needs to know a background region of size back_extraction_radius defind in the config.py. So the process is as follows

- Take an initial point (back_extraction_radius / 2, back_extraction_radius / 2) in the image.
- Find the sky deviation within a region of radius back_extraction_radius. If this deviation is less than the n * sky sigma (where n = 2 as starting value), take that region as background region, else go to the point (back_extraction_radius / 2 + 2.0, back_extraction_radius / 2 + 2.0).
- The above process will go on till it reaches (size - back_extraction_radius / 2 , size - back_extraction_radius / 2) where size is the image size.
- Still the result is negative, increase n from 2 to 3 and continue the process till we find the background region.
- This process has disadvantage as it won't consider the gradient of sky.

The parameters defined in the PyMorph as follows

Goodness

- It is defind as the ratio of number pixels within n times sky sigma around sky value to the total number of pixels.

Flags

The flags used in PyMorph are the following

Explanation	Flag
Repeat Mode	1
Fit bulge center	2
Fit disk center	4
Fit sky	8
The cutimage extend goes outside the image	16
Galaxy ellipse failed	32
Casgm failed	64
Galfit failed	128
Plotting failed	256
Fitting bulge	512
Fitting disk	1024
Fitting point	2048
Neighbour fit	4096
Large chisq	8192
Low goodness	16384
Fake center	32768
Sersic parameter touch limit	65536
Disk parameter touch limit	131072

How to run PyMorph?

1. tar xzvf PyMorph.tar.gz
2. cd PyMorph
3. mv pymorph /your/suitable/area
4. cp pymorph.py config.py /your/data/area/where/you/want/to/run/pymorph
5. Edit .cshrc file and give

   ```
   setenv PYTHONPATH /your/suitable/area/pymorph
   ```
6. cd /your/data/area/where/you/want/to/run/pymorph
7. Edit config.py and add path to your GALFIT and SExtractor binaries.
8. ./pymorph.py

Appendix

As you know 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.

Parameter File

```
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      %(pymorph_path)s/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    <Value from config.py>           # 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  SEGMENTATION    # 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     <Your weight image>    # weight-map filename
WEIGHT_GAIN      N              # modulate gain (E/ADU) with weights? (Y/N)
```

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
ISO0
A_IMAGE
FLAGS
CLASS_STAR
MAG_BEST           Uses in the case of findandfit mode
```

The Convolution Kernel

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

Neural Netwrok

PyMorph uses the default.nnw file coming with SExtractor