中国科学院大学
University of Chinese Academy of Sciences

中国科学院上海天文台
Shanghai Astronomical Observatory, CAS

Galaxy Profile Fitting Package in Big Data Era

Dissertation Defense Presentation

Chen Mi 陈宓

Prof. Dr Shen Shiyin 沈世银 Prof. Dr Rafael S. de Souza 拉斐尔

May 13th, 2024

Outline

Introduction

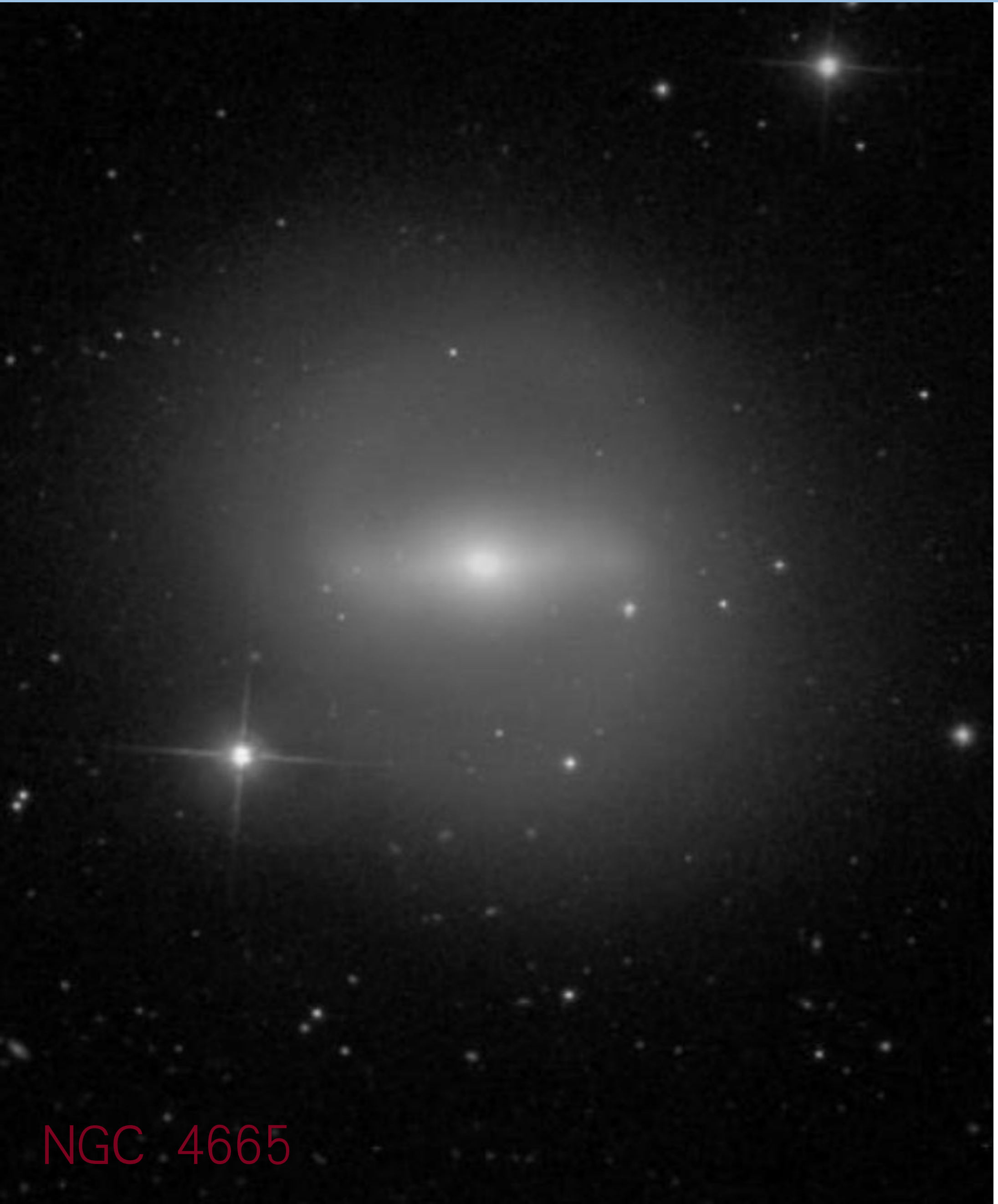
GALMOSS package

- Methods
- Usage
- Evaluation

Discussion&Conclusion

Publications during Master

Introduction



What can we do with a galaxy image?

Visual Classification

Sb galaxy; disk, bulge, bar and a weak spiral arm……

Non-parametric morphological analysis

employs **quantitative metrics** to represent various **attributes** (Concentration, Asymmetry, and Smoothness …)

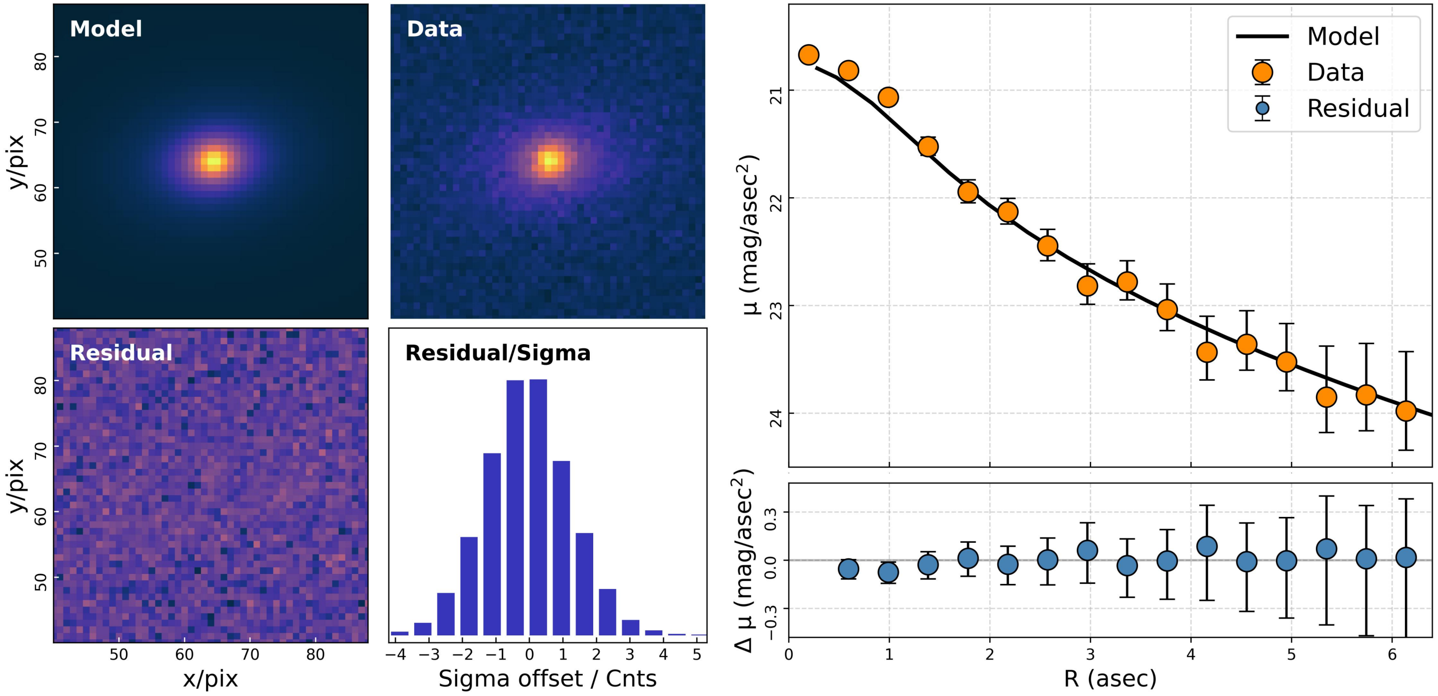
Surface brightness profile fitting

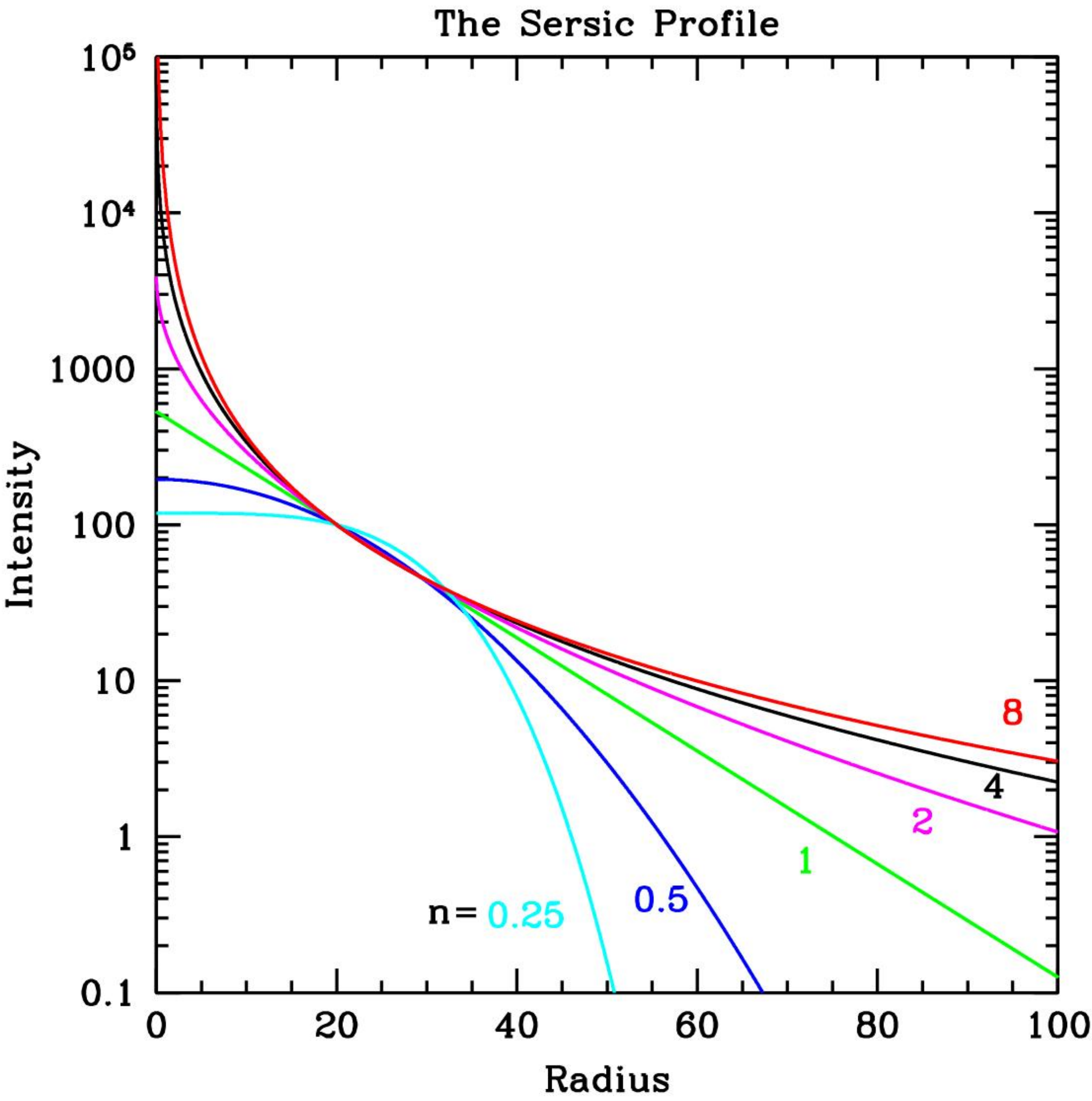
models the distribution of galay flux

Why we do profile fitting?

Let's take Sersic profile as an example:

$$I(r) = I_e \exp\left\{-\nu_n\left[\left(\frac{r}{r_e}\right)^{\frac{1}{n}} - 1\right]\right\}$$

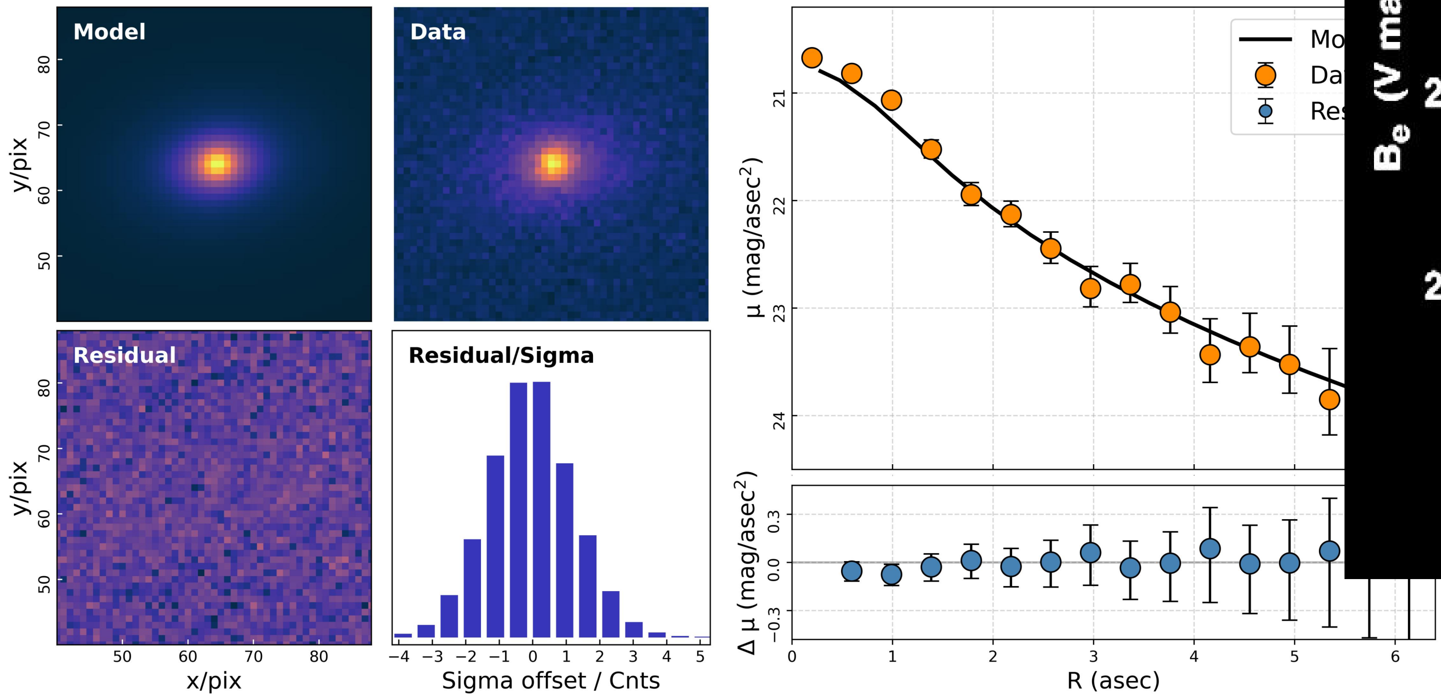


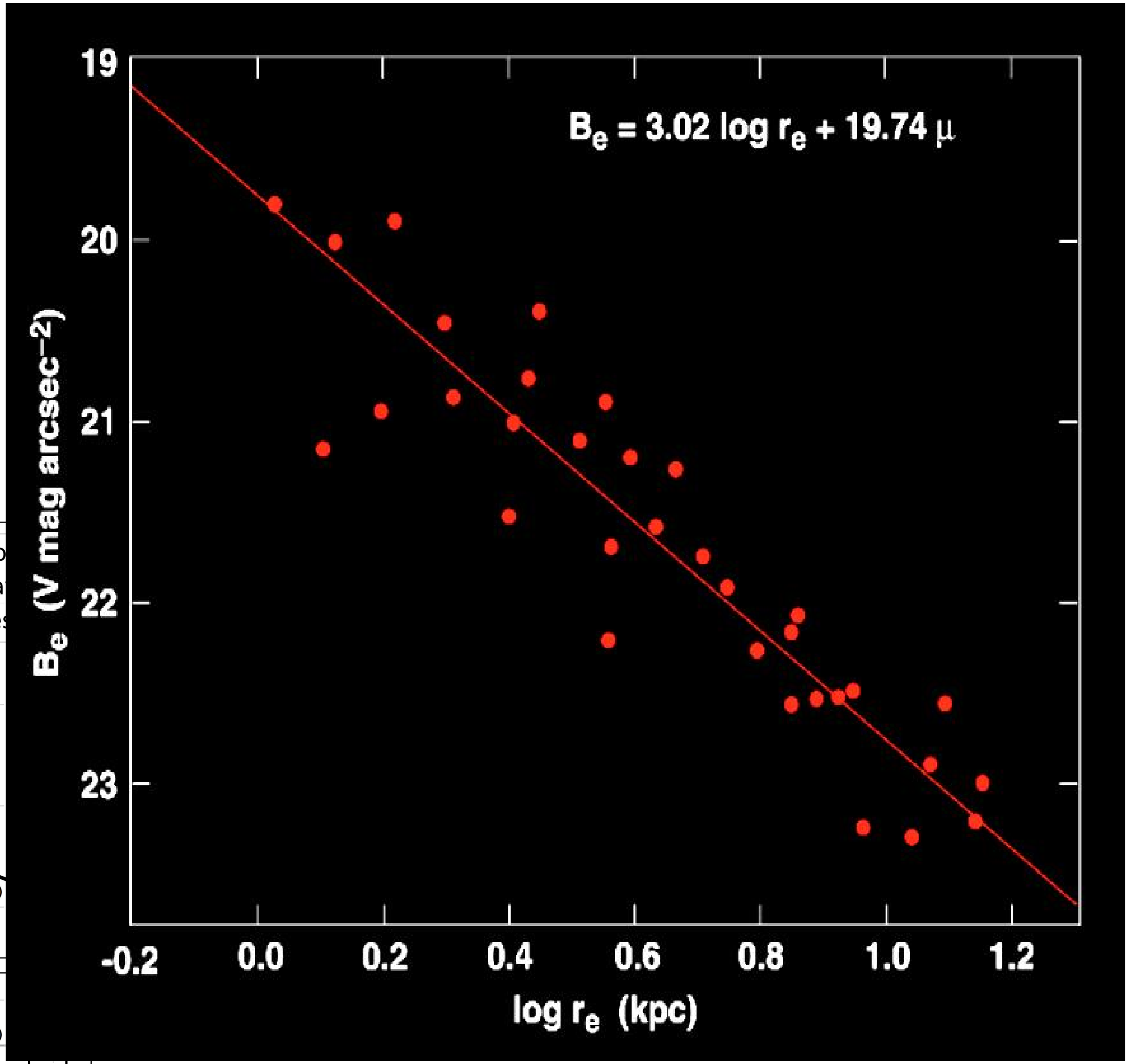


Why we do profile fitting?

Let's take Sersic profile as an example:

$$I(r) = I_e \exp\left\{-\nu_n\left[\left(\frac{r}{r_e}\right)^{\frac{1}{n}} - 1\right]\right\}$$

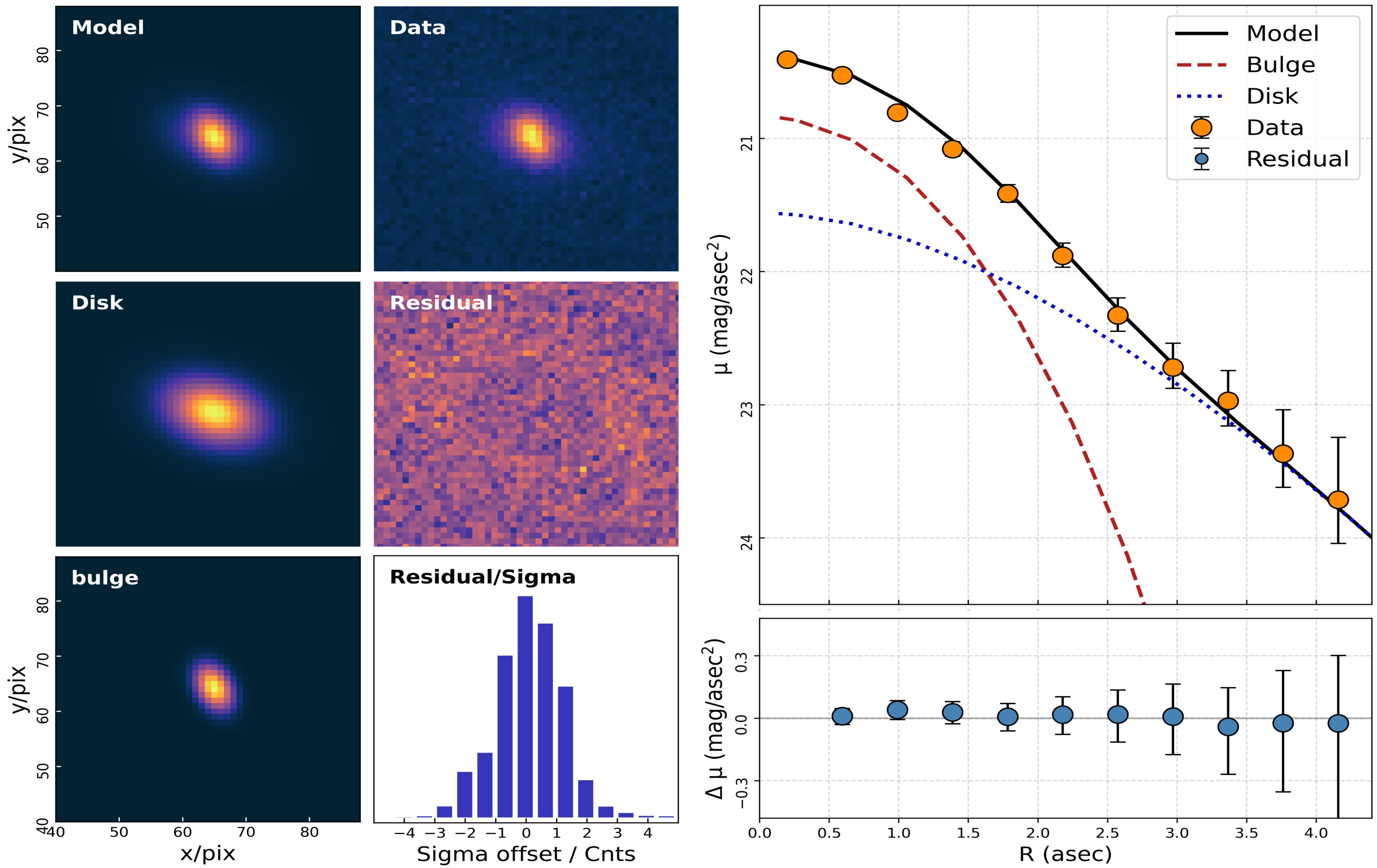


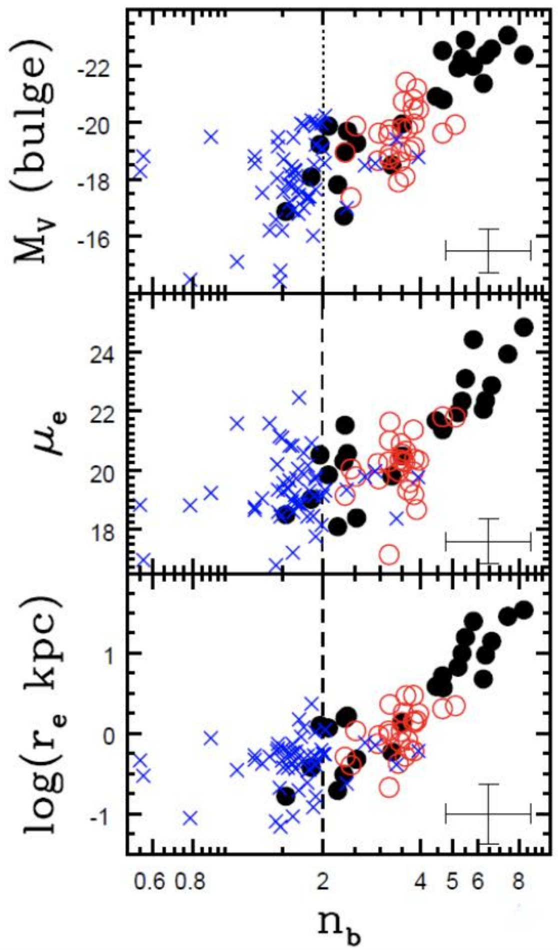


Kormendy (1977)

Why we do profile fitting?

Let's take decomposition with multi-components as an example:





(Fisher&Drory, 2008)

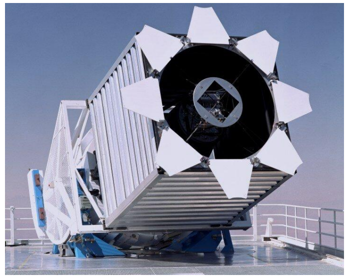
The previous works on this field

Package	open-source	support user-defined profiles	methods	likelihood
GALFIT	×	×	Levenberg-Marquardt (LM)	χ^2 distribution
IMFIT	✓	✓	LM Nelder-Mead Simplex Differential Evolution	χ^2 distribution Poission distribution
PROFIT	✓	✓	MCMC gradient optimizer genetic algorithm	χ^2 distribution Poission distribution normal distribution Student-T distribution
......				

Surveys and big data era

2MASS(1990)
~10TB

SDSS(2000)
~30TB



PanSTARRS(2010)
~10 PB

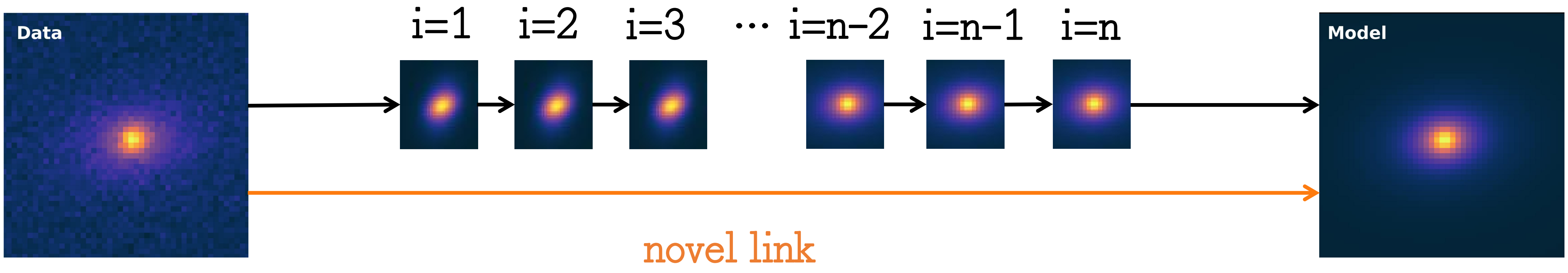
Euclid(2023)
~100PB

CSST(?)

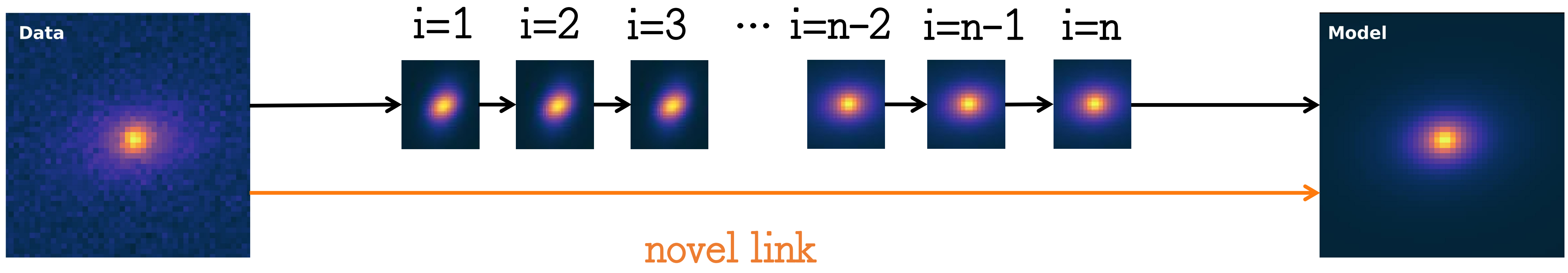
……

Deep Learning

train a **novel link** using batches of galaxies

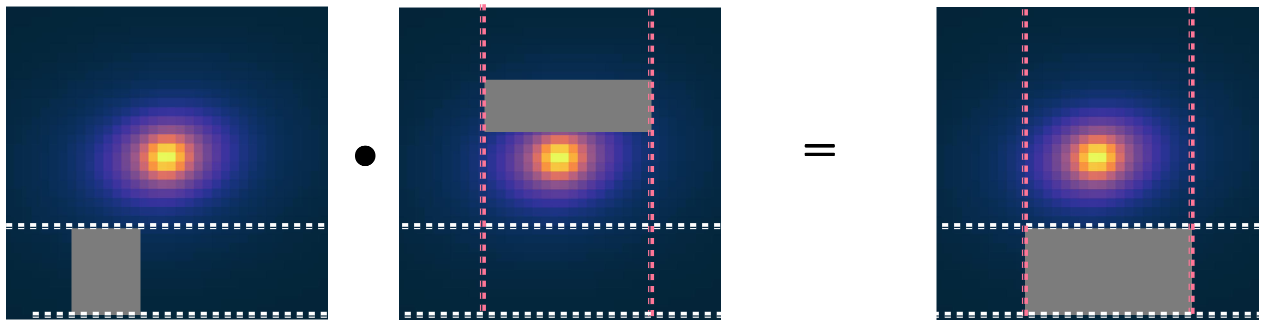


Deep Learning train a **novel link** using batches of galaxies



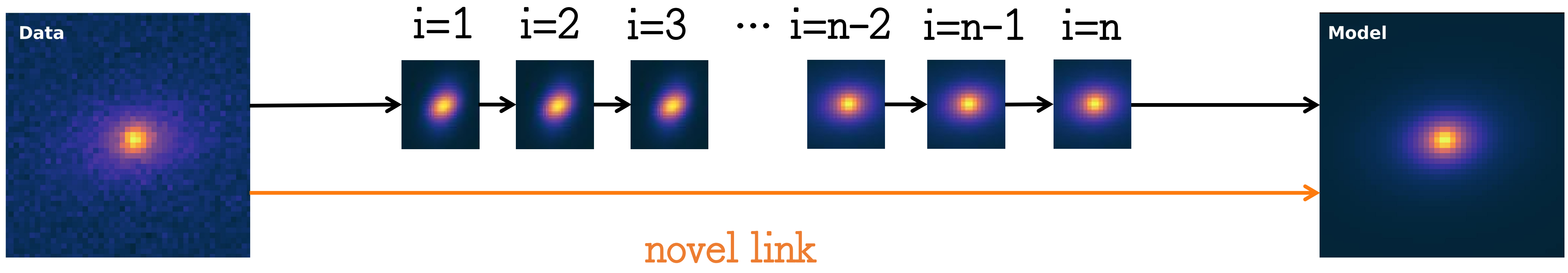
GPU & PyTorch speed up the training process & simplify the coding process

Split the matrix and calculate it in parallel.



Deep Learning

train a **novel link** using batches of galaxies

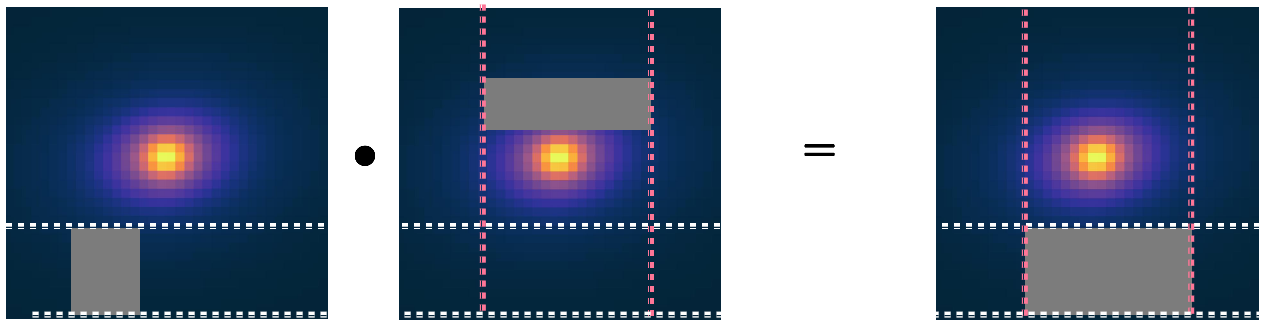


GPU & PyTorch

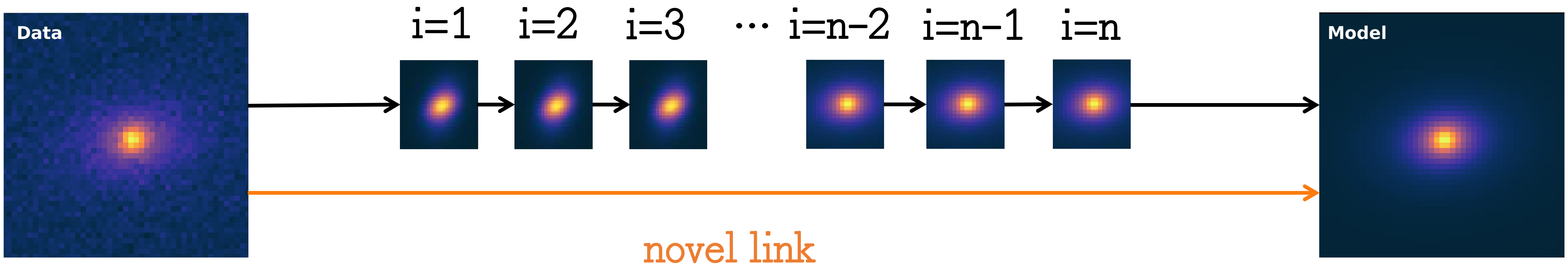
speed up the training process & simplify the coding process

Split the matrix and calculate it in parallel.

Help organizing the split process.



Deep Learning train a **novel link** using batches of galaxies



GPU & PyTorch speed up the training process & simplify the coding process

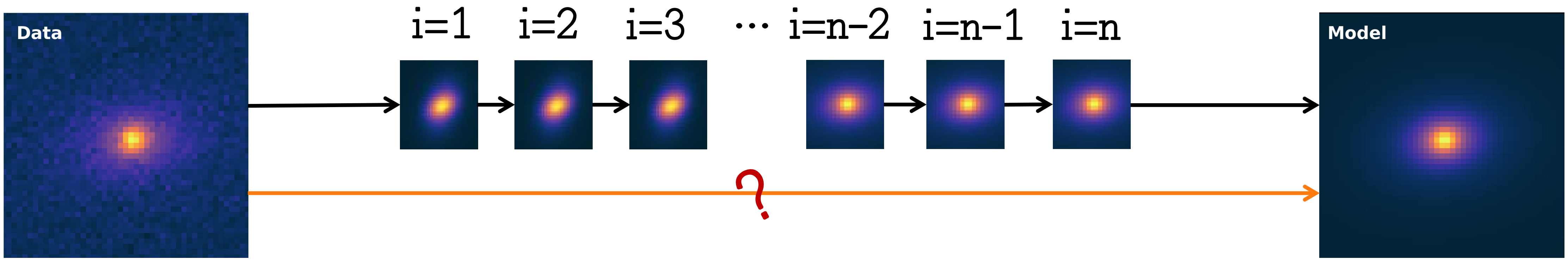
Split the matrix and calculate it in parallel.

Help organizing the split process.

Support various functions for auto-differentiation and parallel calculation

Support various functions for deep learning models

Deep Learning train a **novel link** using batches of galaxies



GPU & PyTorch speed up the training process & simplify the coding process

Split the matrix and calculate it in parallel.

Help organizing the split process.

Support various functions for auto-differentiation and parallel calculation

Support various functions for deep learning models

GIM2D

GalNet

IMFIT

GALFIT

DeepLeGaTo

traditional packages:

intepretable BUT slow

Deep learning models:

fast BUT unintepretable

GIM2D

GalNet

IMFIT

GALFIT

DeepLeGaTo

traditional packages:
intepretable BUT slow

GALMOSS

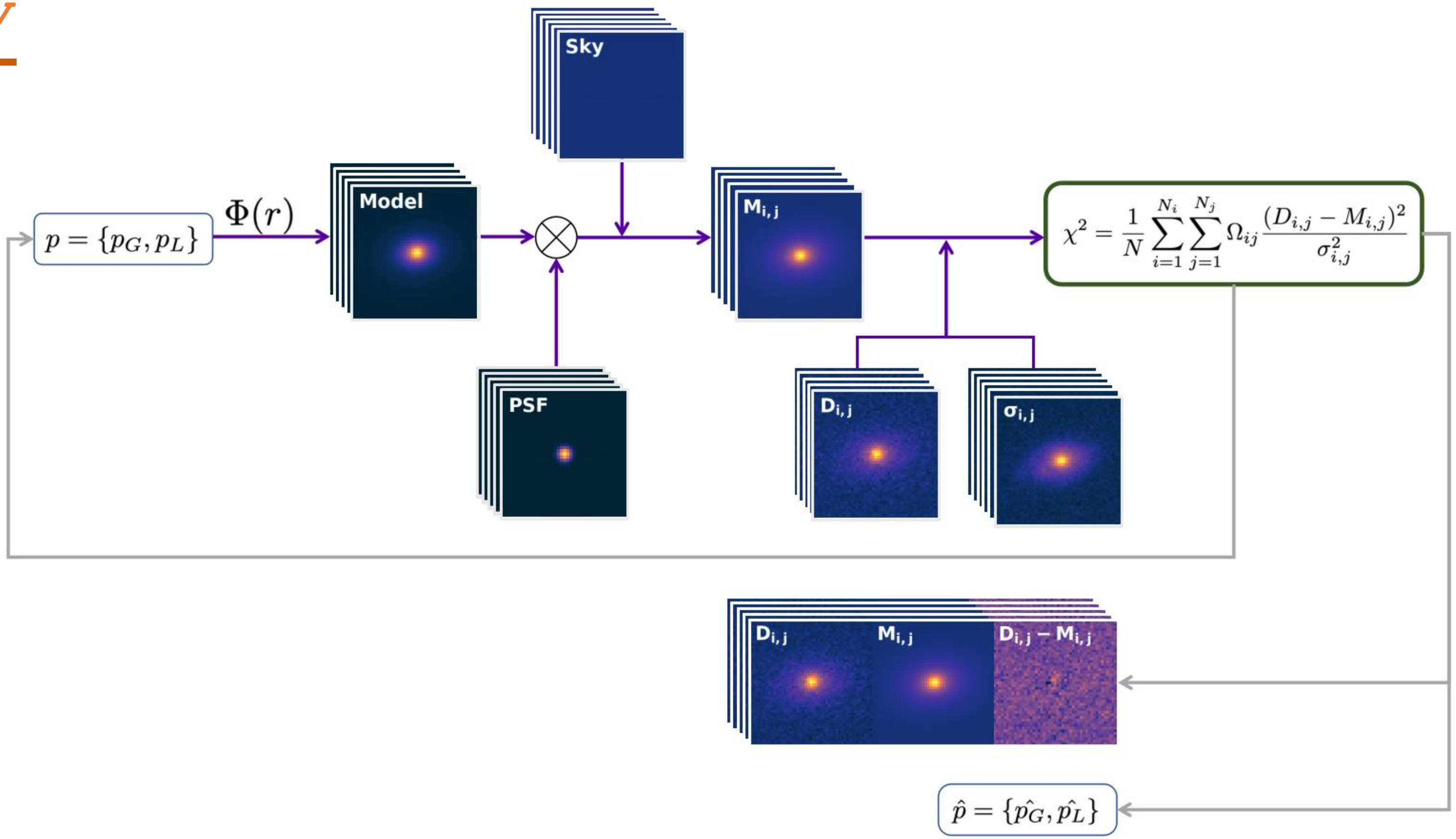
Deep learning models:
fast BUT uninteprepretable

intepretable AND fast

GALMOSS package

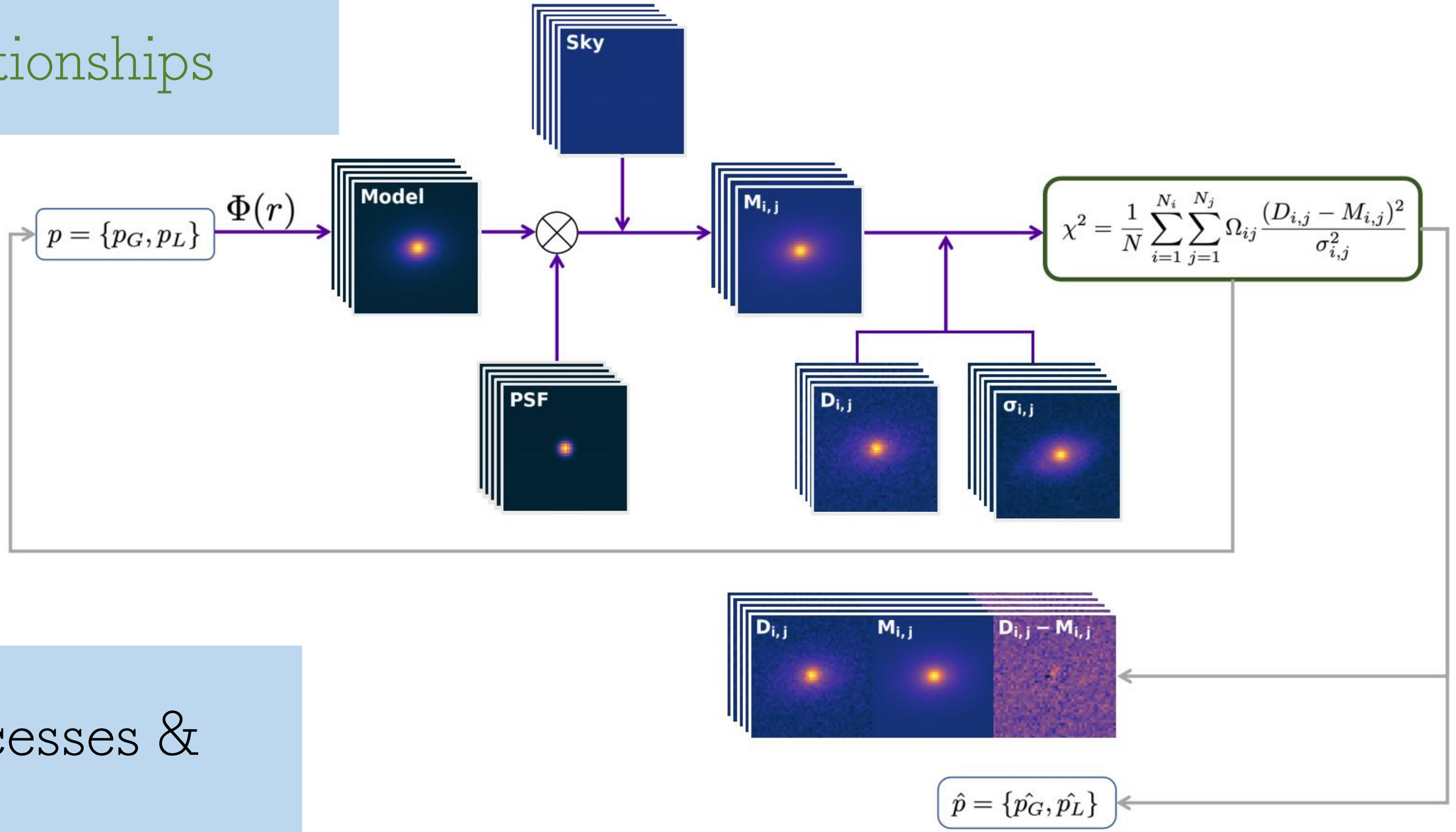
Methods
Usage
Evaluation

Workflow


$p = \{p_G, p_L\}$
$\Phi(r)$
Model
Sky
PSF
$M_{i,j}$
$\chi^2 = \frac{1}{N}\sum_{i=1}^{N_i}\sum_{j=1}^{N_j}\Omega_{ij}\frac{(D_{i,j}-M_{i,j})^2}{\sigma_{i,j}^2}$
$D_{i,j}$
$\sigma_{i,j}$
$D_{i,j}$
$M_{i,j}$
$D_{i,j} - M_{i,j}$
$\hat{p} = \{\hat{p_G}, \hat{p_L}\}$

INTEPRETABLE

maintain physical relationships



FAST

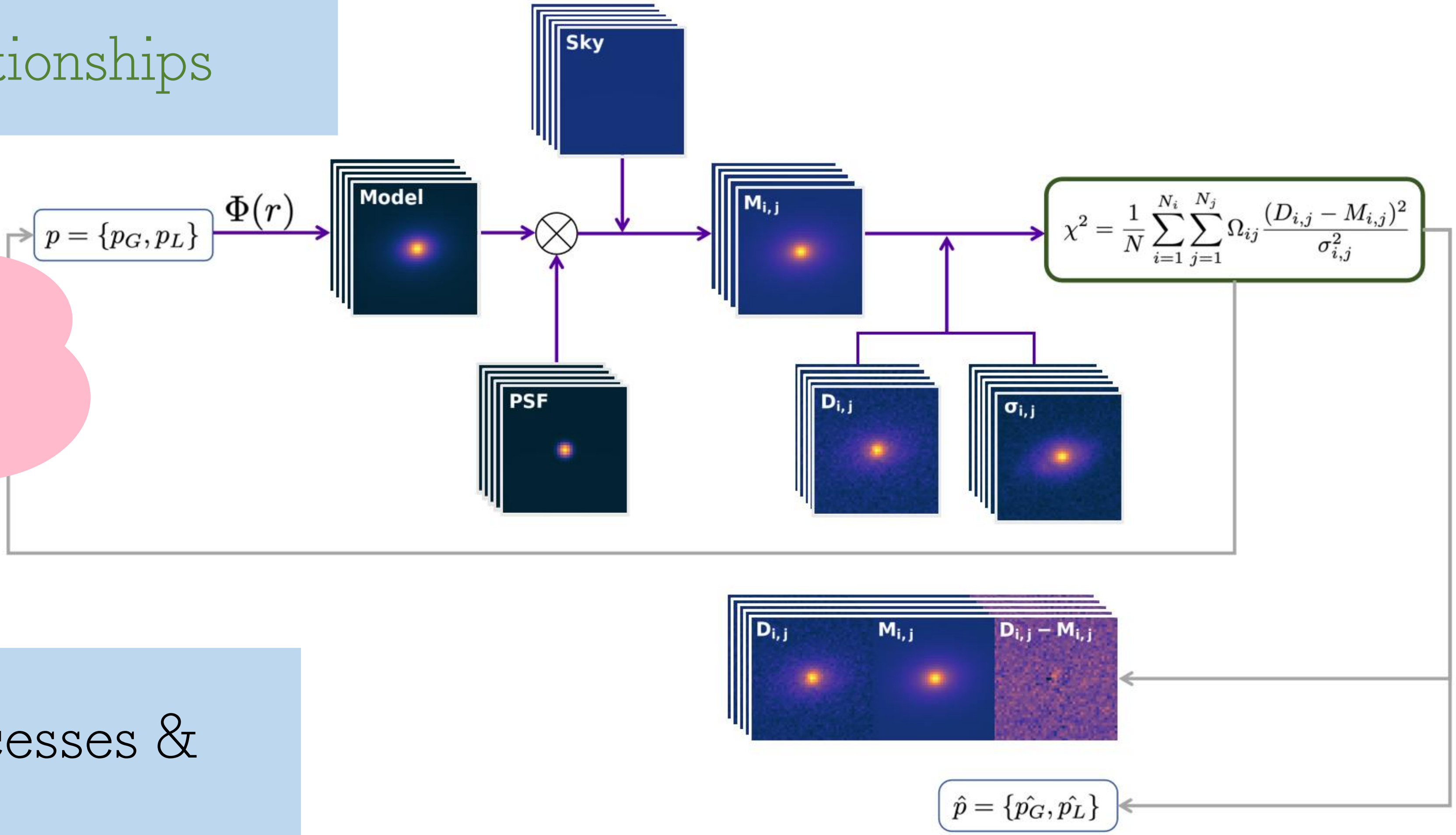
parallelized fitting processes & GPU acceleration

INTEPRETABLE

maintain physical relationships

GPU & PyTorch

speed up the training process
& simplify the coding process

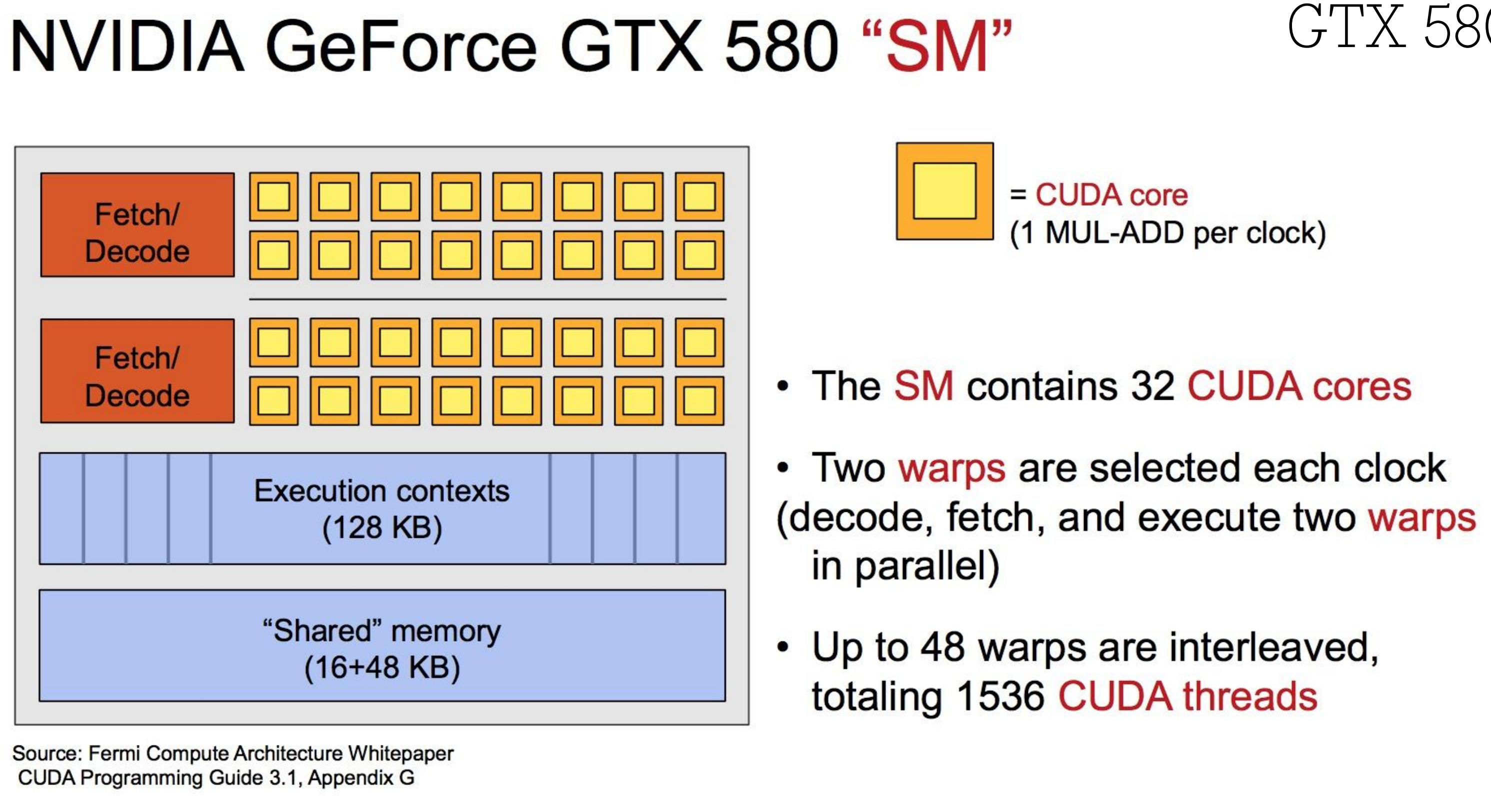


FAST

parallelized fitting processes &
GPU acceleration

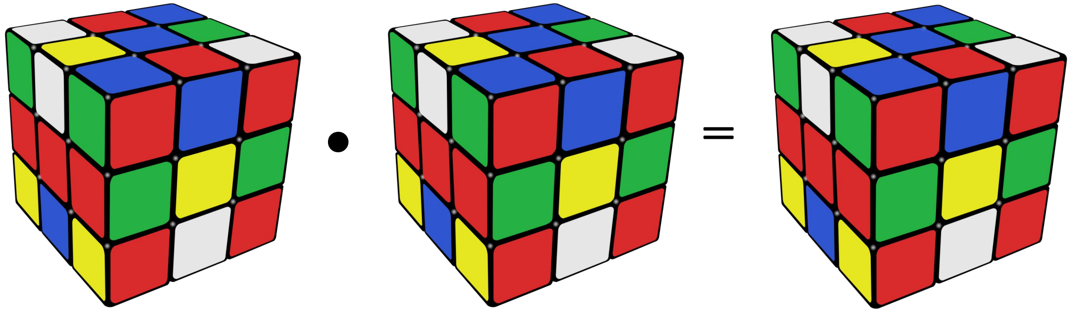
FAST take fully usage of the power of GPU!

GTX 580 has 16 SMs!



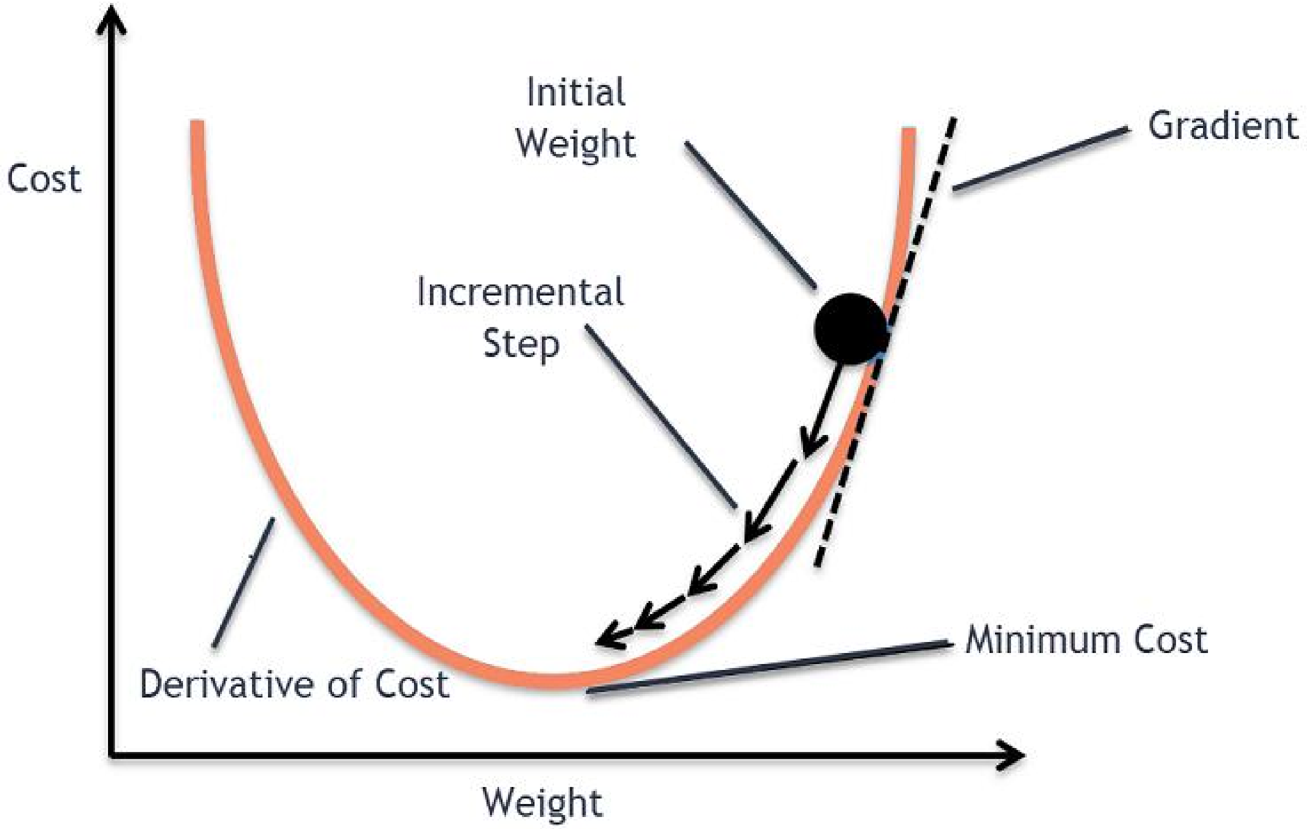
FAST **take fully usage of the power of** GPU!

time of fitting one galaxy = fitting batches of galaxies



fit 1000 galaxies together = 1000 times faster!

Method: gradient descent



$$\theta_{t+1} = \theta_t - \alpha g_t, \quad g_t = \nabla J(\theta_t)$$

Prons

Rapid execution

Low computational and memory demands

--> support big samples;

big images;

multi-params

Auto differentiation supported by PyTorch

--> support combinations of profiles;

easily integration of user-defined profiles

Cons

Cannot estimate fitting uncertainty!

a. covariance matrix

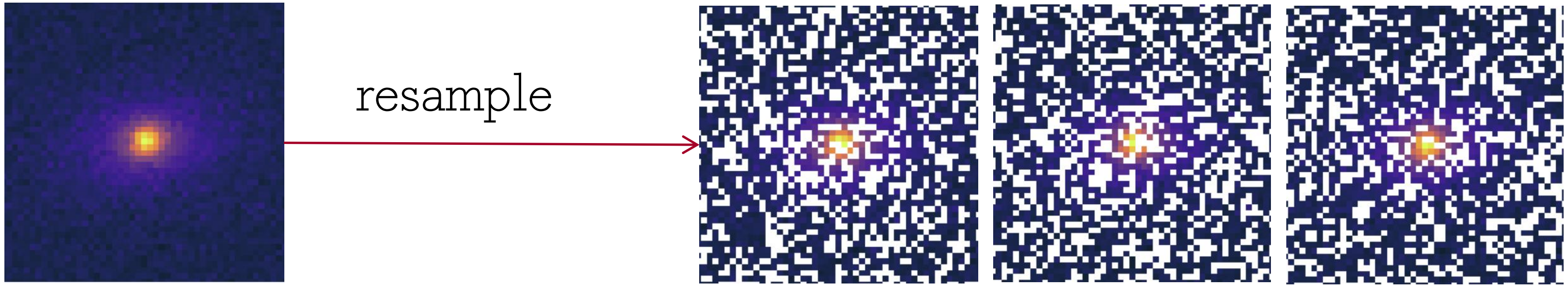
fitting uncertainty consists of observational uncertainty

$$\sigma_{p} = \sqrt{\mathrm{diag}\left(\left[\boldsymbol{J}^{\top}\boldsymbol{W}\boldsymbol{J}\right]^{-1}\right)}$$

$\boldsymbol{J}$	Jacobian matrix
$\boldsymbol{J}^{\top}\boldsymbol{W}\boldsymbol{J}$	Jacobian matrix -> Hessian matrix
$\left[\boldsymbol{J}^{\top}\boldsymbol{W}\boldsymbol{J}\right]^{-1}$	Jacobian matrix -> Hessian matrix-> Covariance matrix

b. bootstrapping

using bootstrapping to resample galaxy images， and use the variance of the re-fit results as fitting uncertainty



$$\sigma^2 = \frac{1}{M}\sum_{i=1}^{M}\left[m_i - \bar{m}\right]^2$$

Basic usage — sersic profile fitting

1. Import galmoss

```
import Galmoss as gm
```

Basic usage — sersic profile fitting (tips: python-based)

1. Import galmoss

```
import Galmoss as gm
```

2. Define profiles

```
# define parameter objects and profile

sersic = gm.lp.Sersic(
    cen_x=gm.p(65.43),
    cen_y=gm.p(64.95),
    pa=gm.p(-81.06, angle=True),
    axis_r=gm.p(0.64),
    eff_r=gm.p(7.58, pix_scale=0.396),
    ser_n=gm.p(1.53, log=True),
    mag=gm.p(17.68, M0=22.5)
)
```

Basic usage — sersic profile fitting

1. Import galmoss

2. Define profiles

3. Define dataset

```
import Galmoss as gm

# define parameter objects and profile

sersic = gm.lp.Sersic(

dataset = gm.Dataset(
    galaxy_index="J162123.19+322056.4",
    image_path="./J162123.19+322056.4_image.fits",
    sigma_path="./J162123.19+322056.4_sigma.fits",
    psf_path="./J162123.19+322056.4_psf.fits",
    mask_path="./J162123.19+322056.4_mask.fits"
    mask_index=2,
    img_block_path="./test_repo",
    result_path="./test_repo"
)

dataset.define_profiles(sersic=sersic)
```

Basic usage — sersic profile fitting

1. Import galmoss
2. Define profiles
3. Define dataset
4. Start fitting
5. Estimate fitting uncertainty

```
import Galmoss as gm

# define parameter objects and profile

sersic = gm.lp.Sersic(

dataset = gm.Dataset(
    galaxy_index="J162123.19+322056.4",
    image_path="./J162123.19+322056.4_image.fits",
    sigma_path="./J162123.19+322056.4_sigma.fits",
    psf_path="./J162123.19+322056.4_psf.fits",
    mask_path="./J162123.19+322056.4_mask.fits"
    mask_index=2,
    img_block_path="./test_repo",
    result_path="./test_repo"
)

dataset.define_profiles(sersic=sersic)

fitting = gm.Fitting(dataset=dataset,
                     batch_size=1,
                     iteration=1000)
fitting.fit()
fitting.uncertainty(method="covar_mat")
```

Basic usage — bulge+disk decomposition

1. Import galmoss

```
import Galmoss as gm
```

Basic usage — bulge+disk decomposition

1. Import galmoss

```
import Galmoss as gm
```

2. Define shared params

```
xcen = gm.p([65.97, 65.73])
ycen = gm.p([65.30, 64.81])
```

Basic usage — bulge+disk decomposition

1. Import galmoss

```
import Galmoss as gm
```

2. Define shared params

```
xcen = gm.p([65.97, 65.73])
ycen = gm.p([65.30, 64.81])
```

3. Define profiles

```
bulge = gm.lp.Sersic(cen_x=xcen,
                     cen_y=ycen,
                     pa=gm.p([58.7, -8.44], angle=True),
                     axis_r=gm.p([0.75, 0.61709153]),
                     eff_r=gm.p([4.09, 18], pix_scale=0.396),
                     ser_n=gm.p([4, 4], log=True),
                     mag=gm.p([17.97, 15.6911], M0=22.5))

disk = gm.lp.Sersic(cen_x=xcen,
                    cen_y=ycen,
                    pa=gm.p([58.7, -8.44], angle=True),
                    axis_r=gm.p([0.75, 0.61709153]),
                    eff_r=gm.p([4.09, 18], pix_scale=0.396),
                    ser_n=gm.p([1, 1], log=True),
                    mag=gm.p([17.97, 15.6911], M0=22.5))
```

Basic usage — bulge+disk decomposition

1. Import galmoss

```
import Galmoss as gm
```

2. Define shared params

```
xcen = gm.p([65.97, 65.73])
ycen = gm.p([65.30, 64.81])
```

3. Define profiles

```
bulge = gm.lp.Sersic(cen_x=xcen,
```

```
disk = gm.lp.Sersic(cen_x=xcen,
```

4. Define datset

```
dataset = gm.DataSet(["J100247.00+042559.8", "J092800.99+014011.9"],
                     image_path=["./J100247.00+042559.8_image.fits",
                                 "./J092800.99+014011.9_image.fits"],
                     sigma_path=["./J100247.00+042559.8_sigma.fits",
                                 "./J092800.99+014011.9_sigma.fits"],
                     psf_path=["./J100247.00+042559.8_psf.fits",
                               "./J092800.99+014011.9_psf.fits"],
                     mask_path=["./J100247.00+042559.8_mask.fits",
                               "./J092800.99+014011.9_mask.fits"],
                     img_block_path="./test_repo/",
                     result_path="./test_repo/"
)
dataset.define_profiles(bulge=bulge, disk=disk)
```

Basic usage — bulge+disk decomposition

1. Import galmoss

```
import Galmoss as gm
```

2. Define shared params

```
xcen = gm.p([65.97, 65.73])
ycen = gm.p([65.30, 64.81])
```

3. Define profiles

```
bulge = gm.lp.Sersic(cen_x=xcen,
disk = gm.lp.Sersic(cen_x=xcen,
```

4. Define datset

```
dataset = gm.DataSet(["J100247.00+042559.8", "J092800.99+014011.9"],
dataset.define_profiles(bulge=bulge, disk=disk)
```

5. Fitting &
   Estimate fitting uncertainty

```
fitting = gm.Fitting(dataset=dataset,
                     batch_size=1,
                     iteration=1000)
fitting.fit()
fitting.uncertainty(method="bstrap")
```

Further usage — user-defined profiles

2-d profiles: take new sky profile as example

$$I = I_0 + k_x(x - x_0) + k_y(y - y_0)$$

1. Import galmoss

2. Define new profiles:
   - define params
   - define image functions

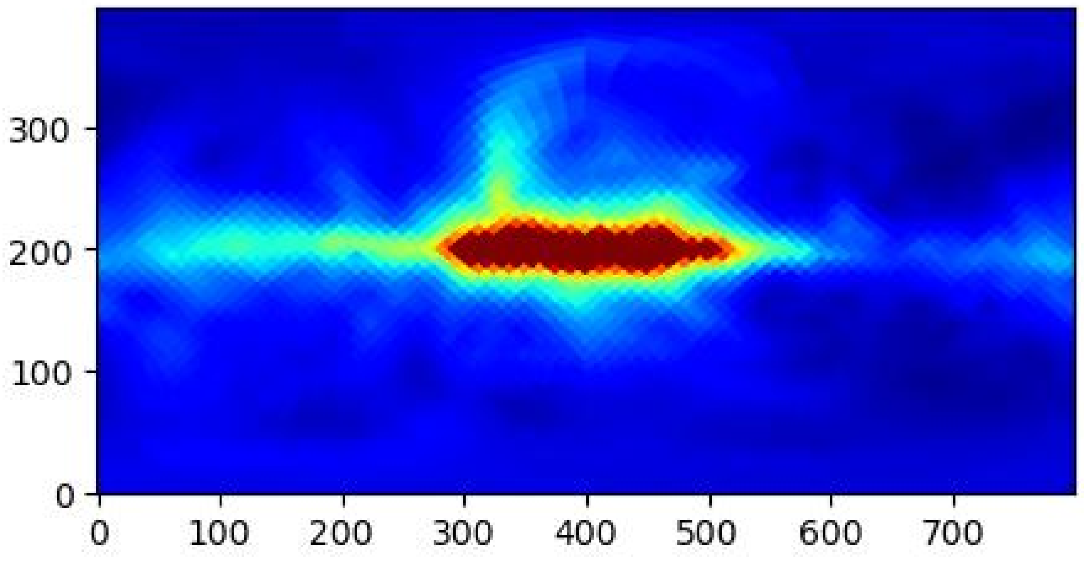
3. The usage of new profiles are the same with built-in profiles

```
import galmoss as gm

class NewSky(gm.LightProfile):
    def __init__(self, sky_0, grad_x, grad_y):
        super().__init__()
        self.psf = False
        self.sky_0 = sky_0
        self.grad_x = grad_x
        self.grad_y = grad_y

    def image_via_grid_from(self,
                            grid,
                            mode="updating_model"):
        return (self.sky_0.value(mode)
                + self.grad_x.value(mode)
                * (grid[1]-(grid[0].shape[1] + 1)/2)
                + self.grad_y.value(mode)
                * (grid[0]-(grid[0].shape[0] + 1)/2)
    )
```

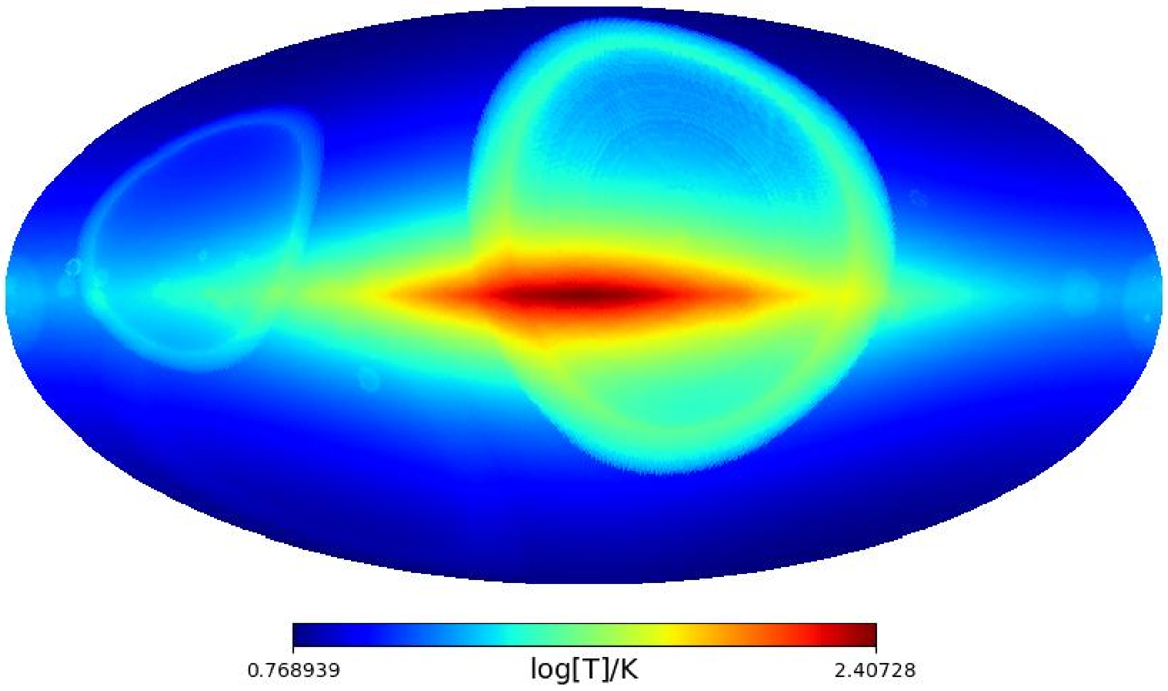
Further usage — user-defined profiles

3-d profiles

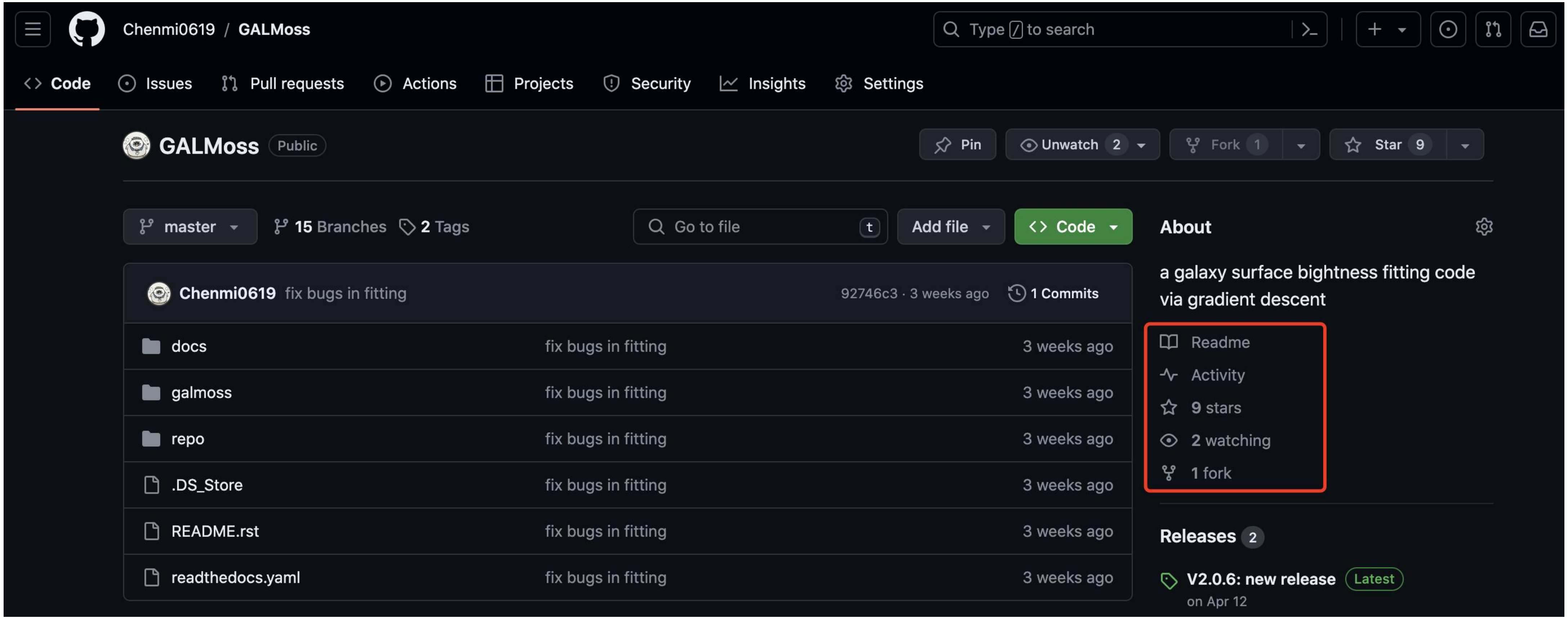
Cong: 3D emissivity fitting (more than 1000 free parameters)



3D emissivity integreated to 2D



Open source pacakge



Open source pacakge

galmoss 2.0.9

```
pip install galmoss
```

A Python-based, Torch-powered tool f
GalMOSS meets the high computatior

Install via pip

```
pip install galmoss
```

Install via git

```
git clone https://github.com/Chenmi0619/GALMoss
cd galmoss
python setup.py install
```

April 19, 2024 (2.0.9) Software Open

View Edit

Chenmi0619/GALMoss: V2.0.9: new release

Chenmi0619

Fix some bugs!

Uploaded on April 19, 2024

45 4

Open source pacakge

galmoss
latest
Search docs

INSTALLATION:
How to install

TUTORIALS:
An easiest start
Combination profiles
User defined profile

MODULES:
galmoss

Read the Docs v: latest

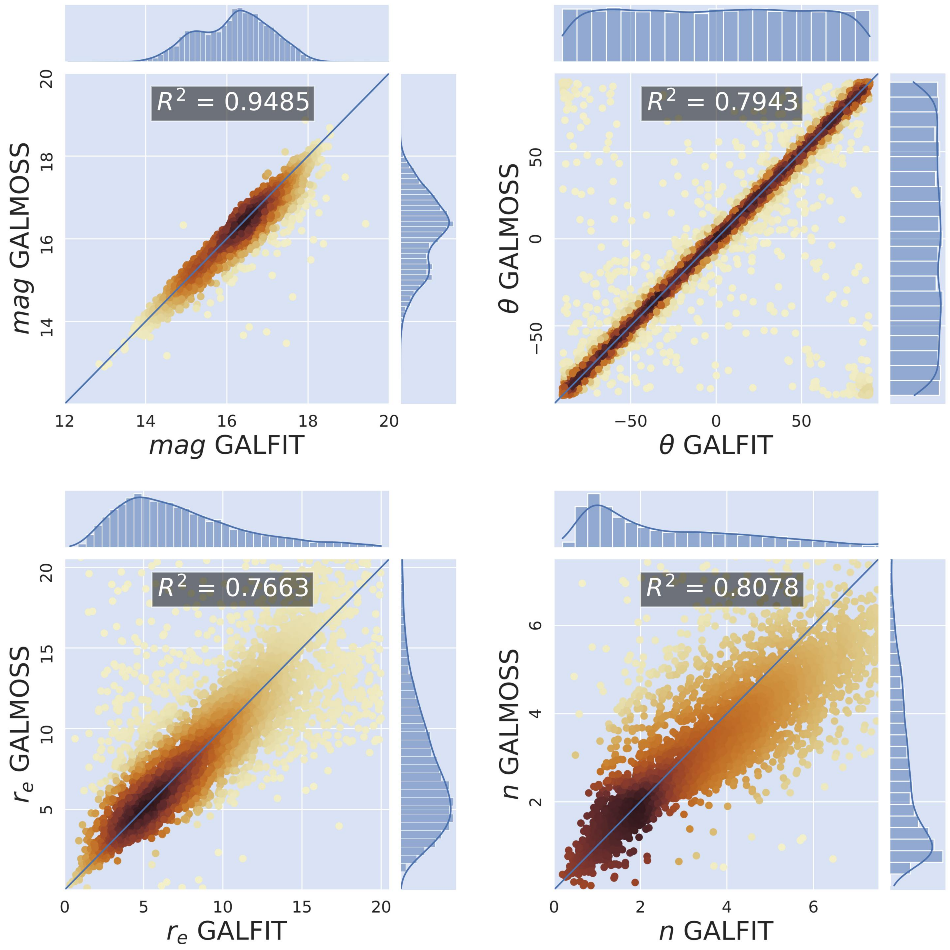
/ Why use GalMOSS? Edit on GitHub

Why use GalMOSS?

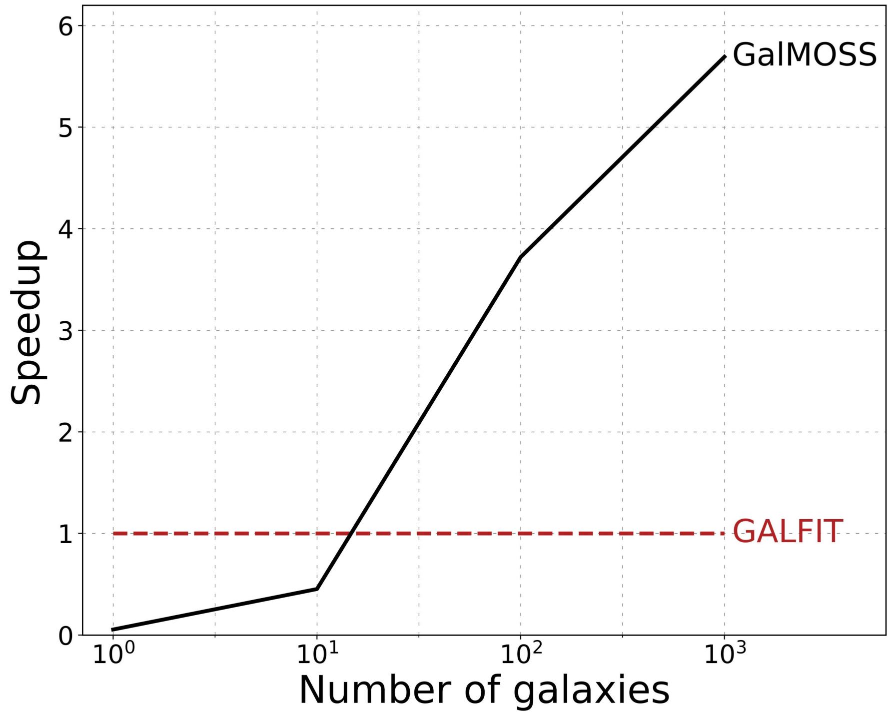
Fit 8,000 galaxies in just 10 minutes!

GalMOSS a Python-based, Torch-powered tool for two-dimensional fitting of galaxy profiles. By seamlessly enabling GPU parallelization, **GalMOSS** meets the high computational demands of large-scale galaxy surveys, placing galaxy profile fitting in the CSST/LSST-era. It incorporates widely used profiles such as the Sérsic, Exponential disk, Ferrer, King, Gaussian, and Moffat profiles, and allows for the easy integration of more complex models.

Evaluation



8324 128*128 SDSS g-band galaxy images
Single sersic
Initial value from SExtractor
in 10 mins
6 times fatser then galfit
(A100& 2.2GHz Intel Xeon Silver 4210)
Compared with MPP-VAC (galfit)

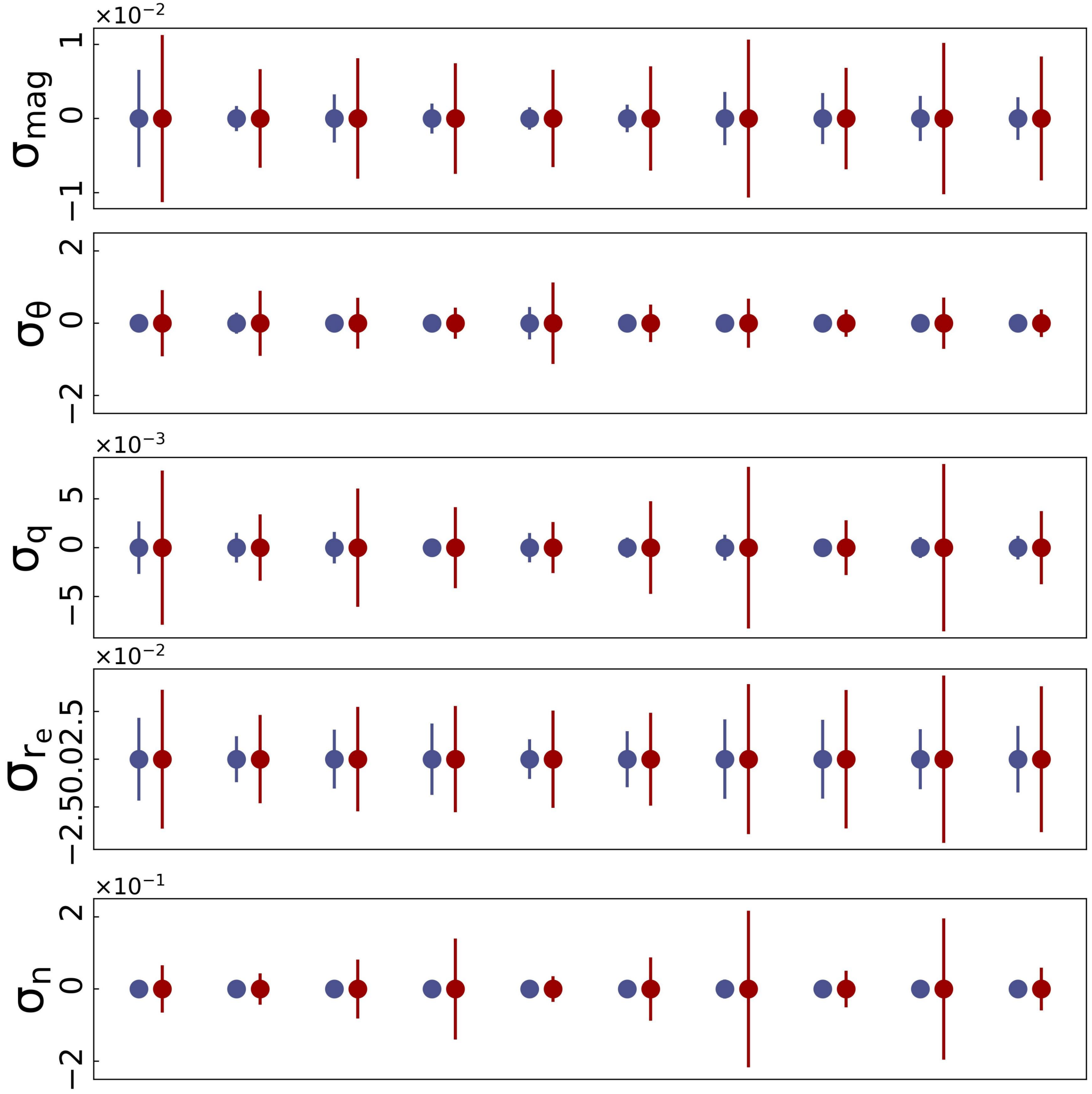


uncertainty comparison

bootstrapping: observation uncertainty;
model uncertainty;
......

larger than

covariance matrix: observation uncertainty



Discussion&Conclusion

Shortcomings

The requirement of human intervention:

provide initial value of parameters

provide image sets
data, psf, sigma, (mask)

fitting

results for further research

galmoss

Human intervention

Shortcomings

The requirement of human intervention: deep learning

~~provide initial value of parameters~~

provide image sets
data, psf, sigma, ~~(mask)~~

fitting

results for further research

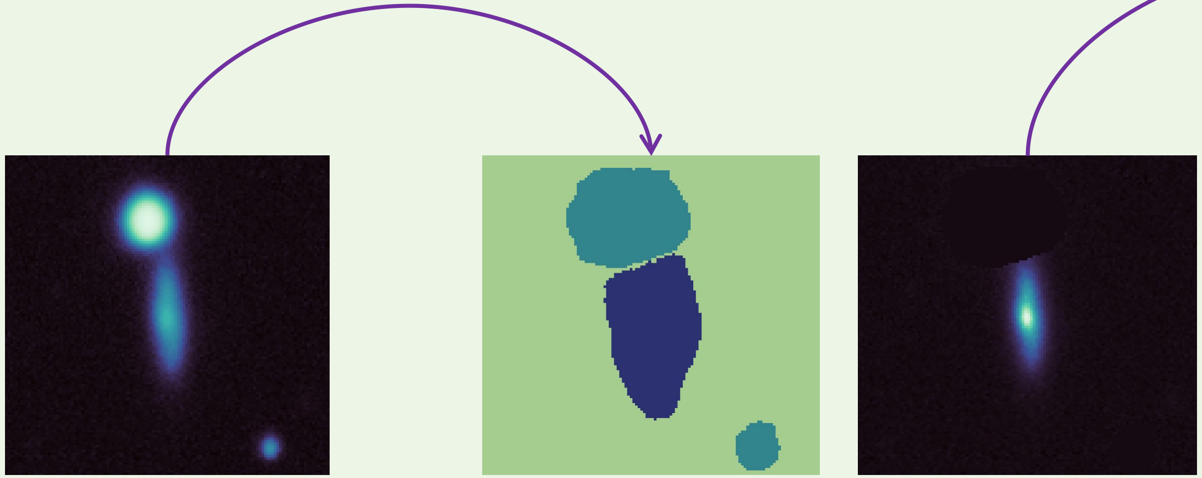
galmoss

Human intervention

Shortcomings

The requirement of human intervention: deep learning

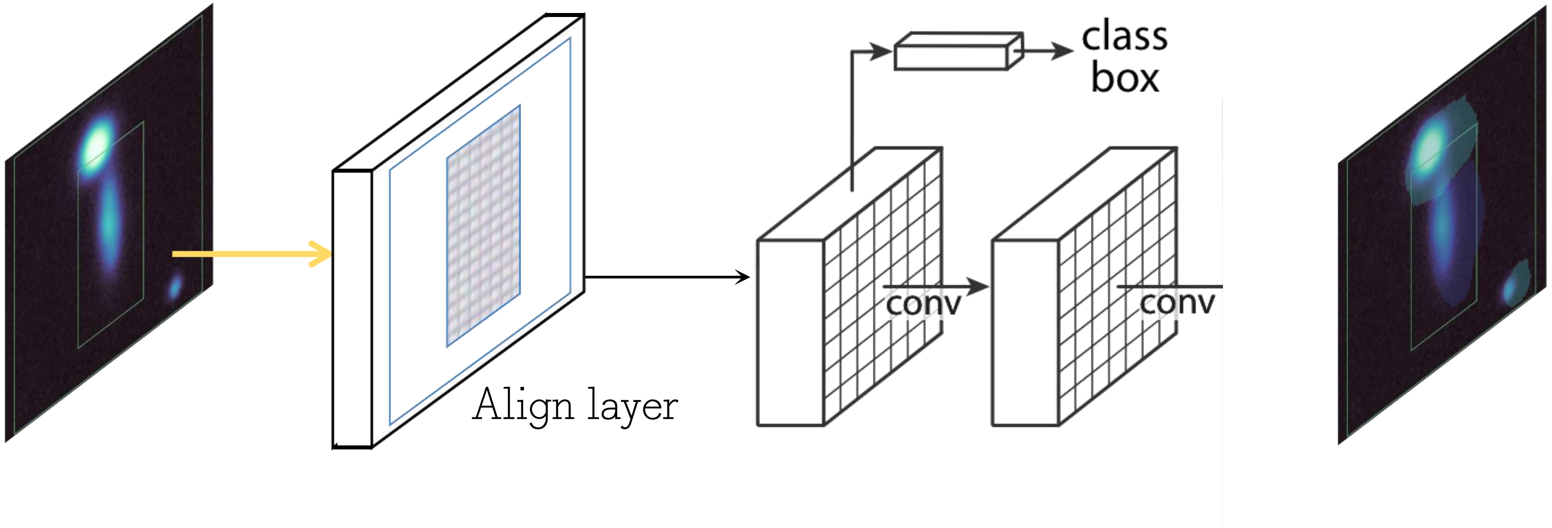
Pre process



·Galaxy center
·Position angle
·Axis ratio
·Magnitude
·Effective radius
·Sersic index

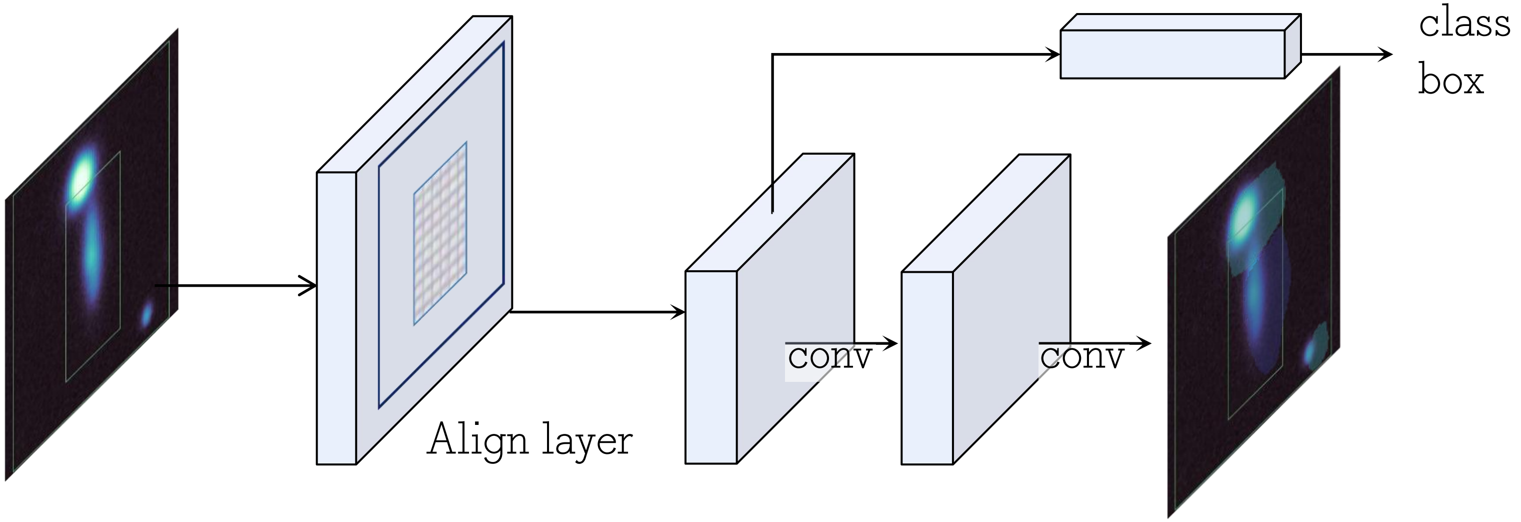
→ GALMOSS

providing mask image: Mask-RCNN



Original image ---> Masked image

- annotate Region Of Interest in the original image, then get feature images with the same size by Align layer
- classify the feature images
- segment the feature images


class
box
Align layer
conv
conv

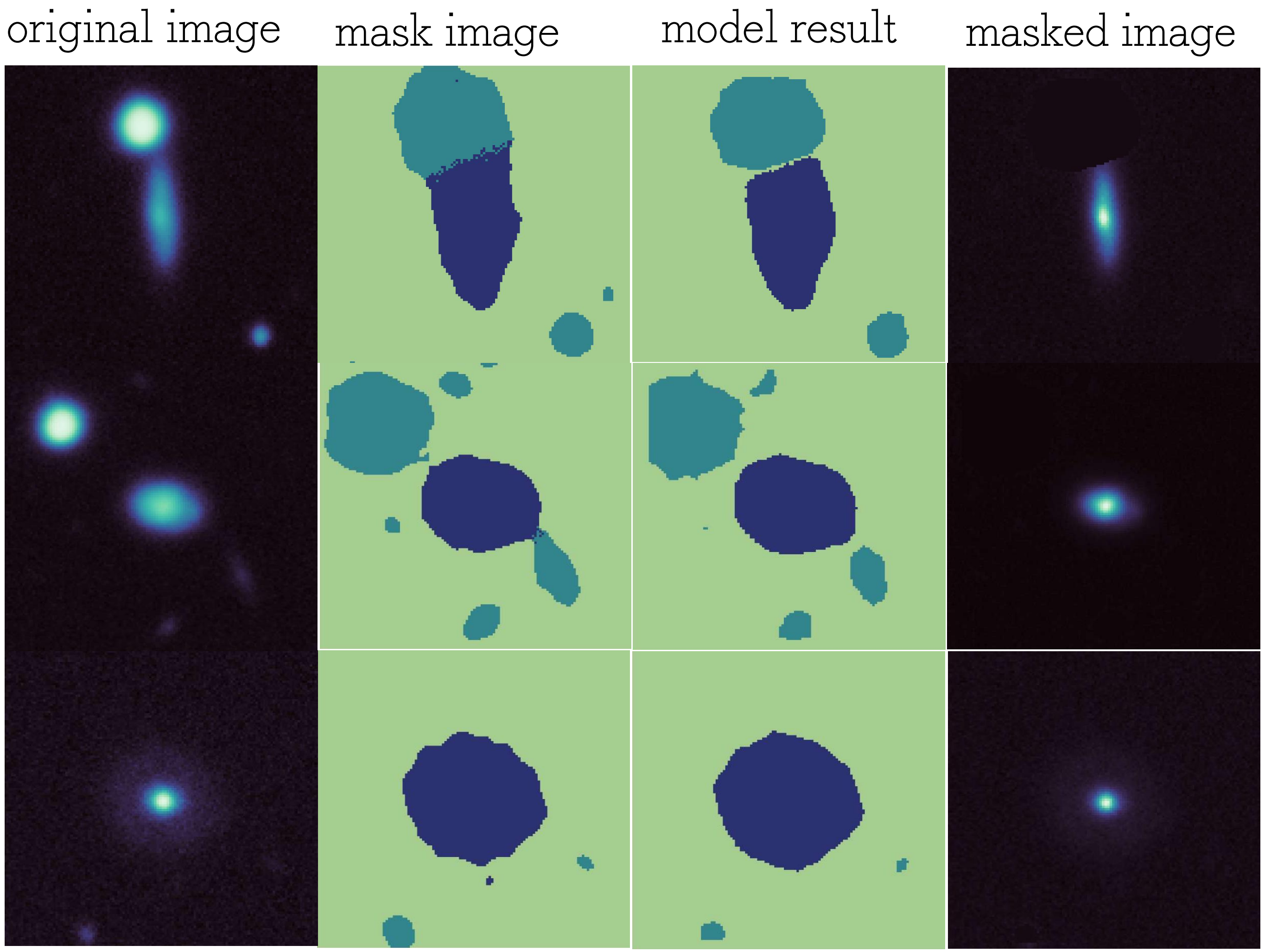
providing mask image: Mask-RCNN

Data:

- DESI image data
- the mask image segmented by SExtractor
- classify the central galaxies as class 2; classify the other objects as class 1

results:

the model segment successfully comparing to the training set.



providing mask image: Mask–RCNN

Data:

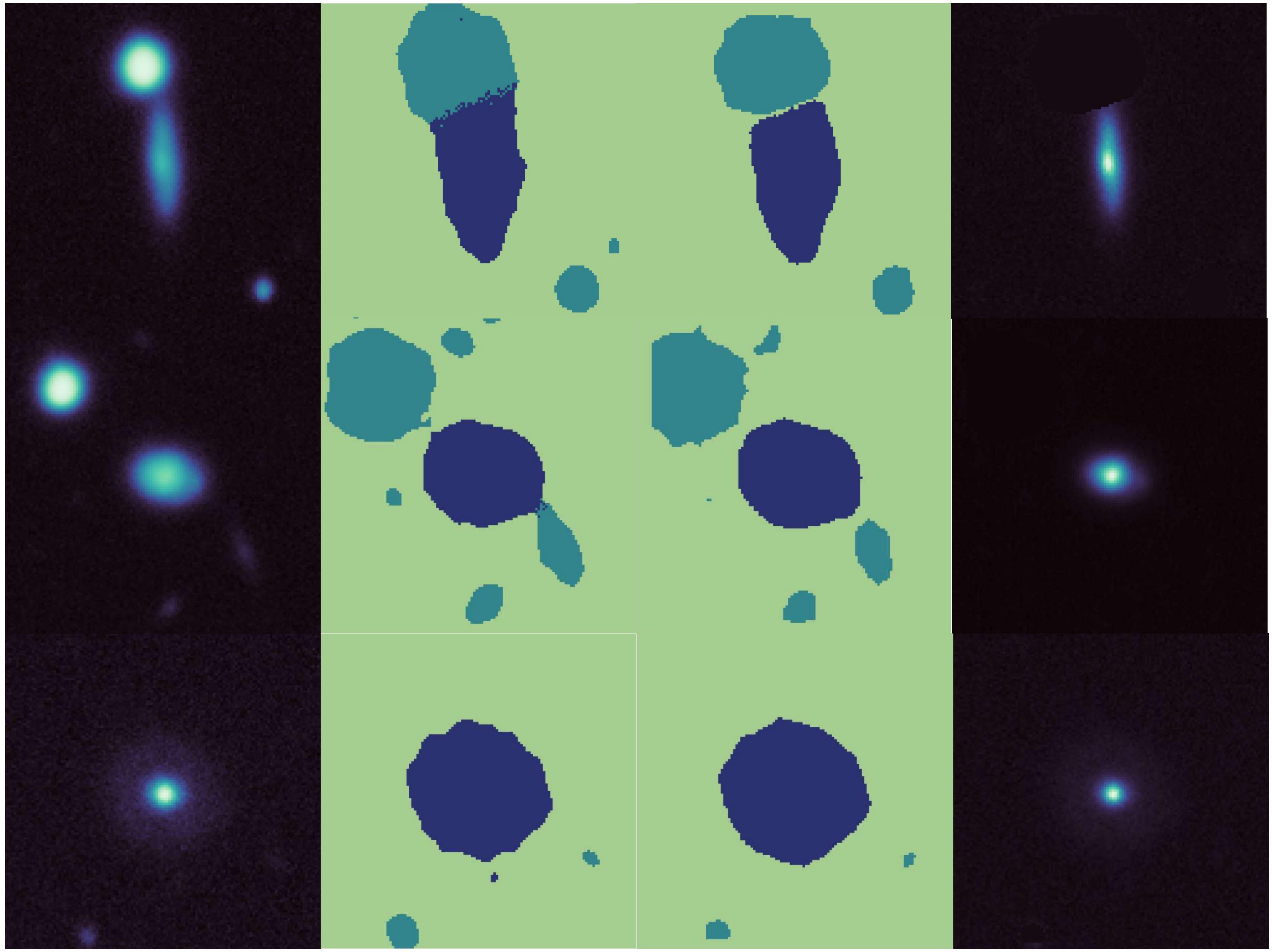
· DESI image data

· the mask image segmented by SExtractor

· classify the central galaxies as class 2; classify the other objects as class 1

results:

the model segment successfully comparing to the training set.



providing initial params: Resnet–50

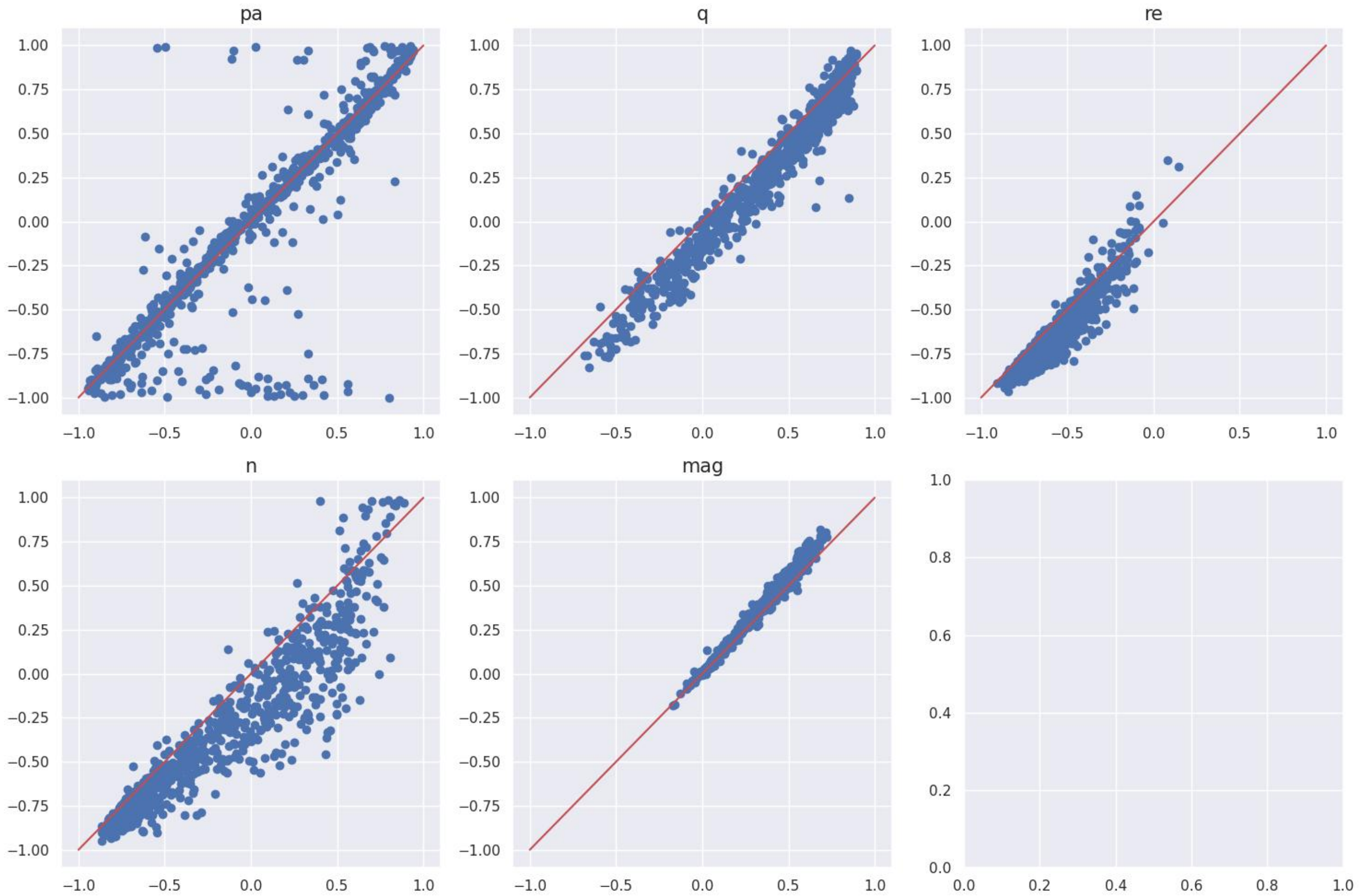
Masked image ---> Initial params

Data:

· DESI image data & galmoss fitting result

results:

the model predict successfully comparing to the training set.



Domain migration: release models for designated servey+size+bands!

conclusion

1. We release an open-source software package galmoss which aims to fit galaxy light profiles on large datasets.
2. galmoss uses gradient descent and χ^2 distribution to fit profiles
3. galmoss speeds up the fitting process by GPU acceleration.
4. galmoss can provide uncertainty through both covariance matrix and bootstrap methods.
5. galmoss also supports the integration of new profiles.
6. Benchmark tests reveal that galmoss is 6 times faster than galfit.

future plans

provide image sets
(image, sigma, psf)

profile fitting with
self-adaption reshaping
under
large sample
large image
large amount of params

results for further research

GALMOSS

Human intervention

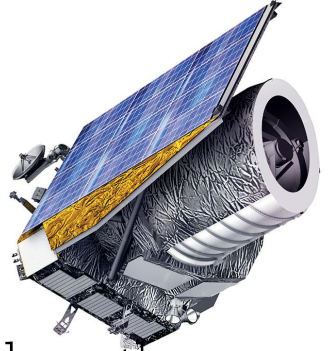
A more automatic package!

publications during Master

1. Chen, M., and de Souza, R. S., Xu, Q., Shen, S., Chies-Santos, A. L., Ye, R., ... Cong, Y.(2024). GalMOSS: GPU-accelerated Galaxy Profile Fitting. Astronomy and Computing, 100825.
2. Xu, Q., and Shen, S., de Souza, R. S., Chen, M., Ye, R., ... Durgesh, R. (2023). From Images to Features: Unbiased Morphology Classification via Variational Auto-Encoders and Domain Adaptation. MNRAS, 526(4), 6391-6400.

Thank you!

APPENDIX



Phd research proposal

Year 1: use measurements of bulge-disc decomposition on Euclid DR1 images to trace how the two components (i.e., bulges and discs) are built up across cosmic history.

Year 2: examine how galaxies of different bulge-to-total ratio b/t populate the star-formation main sequence (SFMS).

Year 3: use the Euclid DR 2 data to classify galaxies into bulge-dominated and disc-dominated types and investigate the dark matter halo masses of these two different morphological types. Test if there is a morphological dependence in the stellar-to-halo-mass relation.

Year 4: use halo assembly history to connect galaxies and therefore to establish a statistical morphological evolution picture. Select broad-line AGNs from the Euclid data and the WEAVE and 4MOST spectroscopic data and obtain SMBH masses for the AGNs.

APPENDIX

Various distributions

Poisson: X-ray astronomy (not enough photons)

T-student: normal distribution but many data points are bad represented by models (e.g., fitting Sersic profiles to well-resolved galaxies with asymmetries)

APPENDIX

Fitting bias

ETGs account for 34.37 % of the total galaxy samples and represent 88.78 % in the biased region.