Astronomy and Computing 47 (2024) 100825

A R T I C L E   I N F O

Keywords: Galaxies, General – methods, Data analysis – methods, Statistical – GPU computing

A B S T R A C T

We introduce 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. Tested on 8289 galaxies from the Sloan Digital Sky Survey (SDSS) g-band with a single NVIDIA A100 GPU, galmoss completed classical Sérsic profile fitting in about 10 min. Benchmark tests show that galmoss achieves computational speeds that are 6 × faster than those of default implementations.

1. Introduction

Galaxies, the cosmic building blocks, comprise diverse stellar components such as bulges, disks, bars, spiral arms, and nuclear star clusters. One of the main drivers of extragalactic astronomy is the study of the structural and morphological properties of galaxies from their photometric images, which has been shown to correlate with the galaxies’ formation and evolutionary paths (van der Wel, 2008; Conselice, 2014; Dimauro et al., 2022).

Several approaches have been developed for galaxy structural and morphological analysis. Classical eyeball morphology classifications, such as the Hubble sequence (Hubble, 1926), are generally descriptive and rely on visual inspection. Non-parametric morphological analysis (Ferrari et al., 2015; Rodriguez-Gomez et al., 2019) employs quantitative metrics to represent attributes such as concentration, asymmetry, and smoothness (Conselice, 2003). These metrics are considered robust because they do not depend on any underlying model. In contrast, the analysis of surface brightness profiles of galaxies involves fitting the light distribution with parametric models to interpret and quantify morphology using specific model parameters (Nantais et al., 2013; Zhuang and Ho, 2022). This profile fitting method has evolved from one-dimensional (de Vaucouleurs, 1958; Sersic, 1968) to two-dimensional analyses, improving accuracy by incorporating factors such as the point spread function and non-axisymmetric components (Andredakis et al., 1995; Schade et al., 1995; Byun and Freeman, 1995).

The fitting of two-dimensional light profiles in galaxies has grown increasingly complex, requiring faster and more scalable algorithms. Two primary methods dominate: Markov Chain Monte Carlo (MCMC) and gradient-based approaches. MCMC-based programs, such as profi t (Robotham et al., 2017) and autogalaxy (Nightingale et al., 2023), iteratively sample to approximate the target distribution, with fitted values and uncertainties derived from the Markov chain’s statistics. Conversely, gradient-based methods, utilized by tools like galfit (Peng et al., 2010) and imfit (Erwin, 2015), offer quicker solutions by directly navigating to the loss function’s local minima. However, this speed compromise the exploration of parameter space and the robustness of uncertainty estimates, a trade-off not present in MCMC-based approaches (Chen et al., 2016).

With the expanding volume of data from astronomical surveys such as the Sloan Digital Sky Survey (SDSS, York et al., 2000), Dark Energy Survey (DES, Abbott et al., 2016), Chinese Survey Space Telescope (CSST, Zhan, 2011), Legacy Survey of Space and Time (LSST, Abell et al., 2009), and EUCLID (Laureijs et al., 2011), significant challenges emerge in galaxy profile fitting tasks. Both gradient-based and MCMC approaches face difficulties in processing the vast number of galaxies efficiently.

To address these challenges, deep learning strategies are increasingly being employed. For instance, deplegato (Tuccillo et al., 2018) and galnets (Li et al., 2022; Qiu et al., 2022) train convolutional neural networks to predict profile-fitting parameters from...
simulated galaxy images. These deep learning approaches establish an
implicit link between galaxy images and their profile parameters,
substantially accelerating the process of parameter estimation (see
also gamornet, Ghosh, 2019). However, these methods cannot be
easily adapted to new data domains due to the well-known ‘black box’
problem (Castelvecchi, 2016), which means that the implicit mappings
generated by the models usually lack clear interpretations.
Another method to address speed challenges involves parallel
gradient-based methodology, which relies exclusively on GPU-accelerated
matrix parallel computation. This is supported by popular frameworks
such as pytorch (Paszke et al., 2019), tensorflow (Abadi et al.,
2016), and jax (Bradbury et al., 2018), enabling the incorporation
of automatic differentiation. Applications of this approach range from
galaxy kinematic modeling (Bekiaris, 2017) to cosmological N-body
simulations (Modi et al., 2021).

To speed up galaxy profile fitting, bringing it to the CSST/LSST-era and
contribute to the literature on galaxy formation and evolution, we
introduce galmoss. We aim to harness the advantages of both tra-
tional and contemporary approaches by providing a framework that
enables seamless parallelization while preserving the interpretability
of more conventional methods. galmoss processes batches of galaxy
photometric images as multidimensional arrays, facilitating efficient
fitting on GPUs. This tensor-based approach permits dynamic man-
agement of data sizes during computations through the adjustment of
batch sizes and the optimization of GPU utilization. For the fitting
process, galmoss makes use of gradient descent due to its relatively
low computational and memory requirements. Finally, galmoss pro-
vides two methods for uncertainty estimation. In the realm of galaxy
profile fitting, astrophot (Stone et al., 2023) leverages GPU accel-
eration and automatic differentiation in its serial optimization process
while preserving physical interpretability. While our package focuses
on processing numerous images of individual galaxies, astrophot is
optimized for fitting multiple galaxies within a single, crowded image.

This paper is structured as follows: Section 2 provides an overview
of the galmoss workflow. Section 3 describes the data that we use
to showcase the software’s performance. Section 4 details the genera-
tion of model images, using the Sérsic profile as an example, and
introduces a set of built-in profiles along with advanced applications.
Section 5 discusses the fitting process post-image generation. Finally,
Section 6 showcases galmoss’ performance through case studies, with
concluding remarks presented in Section 7.

2. General workflow of galmoss

Fig. 1 shows the workflow of galmoss. It begins by reading a
vector \( \mathbf{p} = (\mathbf{p}_i, \mathbf{p}_f) \) consisting of user-defined initial parameter values,
along with the data image, sigma image (data uncertainty), mask im-
age, and Point Spread Function (hereafter PSF) image. The first subset
\( \mathbf{p}_i \) describes the geometric properties, which define the central position

---

1. https://dr15.sdss.org/sas/dr15/eboss/photoObj/
4. Image generation and built-in profiles

For a given set of galaxy or profile parameters, galmoss generates model images that represent the intensity or surface brightness of each pixel, based on the selected profile. galmoss provides six built-in radial profiles: Sérsic, Exponential disk, Ferrer, King, Gaussian, and Moffat, as well as a Flat sky model (with the option to include more complex sky models). These radial profiles are transformed into two-dimensional images by assuming circular symmetry. This process effectively projects the one-dimensional profile over a circular area in two dimensions to create the final image. To illustrate this projection process more concretely, we use the Sérsic profile as an example, given its widespread application in modeling galaxy profiles.

Furthermore, galmoss supports the combination and integration of different profiles. Detailed instructions on the practical use of the code can be found in the Appendices B–D.

4.1. Sérsic

The Sérsic profile (Sérsic, 1968) is commonly used to fit the surface brightness distribution of elliptical galaxies, as well as the disk and bulge components of other galaxy types. The intensity $I(r)$ as a function of the radius $r$ is given by

$$I(r) = I_0 \exp \left(-\frac{n}{r^{1/n}}\right),$$  

where $I_0$ denotes the surface brightness at the effective radius $r_e$, which encompasses half of the total profile flux. The Sérsic index $n$ dictates the profile’s curvature, and the parameter $v_n$ is computed numerically from the inverse cumulative distribution function of the gamma distribution. The versatility of the Sérsic profile lies in its ability to model a wide variety of galaxy profiles, from the de Vaucouleurs to the exponential disks, the boxiness parameter

$$B = \frac{1 - (x - x_c)^2}{(x - x_c)} + \frac{1 - (y - y_c)^2}{(y - y_c)},$$

where $x, y$ are the position coordinates of the ellipse and $x_c, y_c$ are the position angles of the ellipse profile. The axis ratio $q$ and the parameter $s$ are defined in Eq. (6).

The boxiness parameter $B$ introduces flexibility in the shape of the ellipse (Athanassoula et al., 1990), allowing for more general elliptical profiles. $B = 0$ corresponds to a standard ellipse, while positive values result in more box-like shapes and negative values in more disk-like shapes. This flexibility is particularly useful in galaxy modeling, where a range of elliptical shapes can represent the diverse morphologies observed in galaxies.

Rather than directly using the surface brightness $I_e$ in the Sérsic parameter $m$ and its corresponding zero point magnitude $m_0$ is used to specify the intensity level. This approach is consistent with methodologies employed by widely-used galaxy fitting tools such as galfit (Peng et al., 2010) and profit (Robotham et al., 2017), facilitating easier comparison with observational data. Both quantities are related as follows:

$$I_e = \frac{r_{box}}{\pi} \frac{2^{0.3(3m_0-9.9)}}{2xqr^2\Gamma(2n)exp(v_n)},$$

where

$$r_{box} = \frac{x(2 + B)}{2\beta} \left(1 - \frac{1}{2\beta} \right).$$

In this formulation, $r_{box}$ serves as a geometric correction factor in $I_e$ to account for deviations from a perfect ellipse, influenced by the level of diskiness or boxiness. Specifically, when $B = 0$, indicating a perfect ellipse, $r_{box} = 1$, implying no geometric correction. The beta function $\beta(a, b)$ is calculated using the relationship $\beta(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$, where $\Gamma$ denotes the Gamma function.

Fig. 2 presents the fitting results using the Sérsic profile for galaxy J162123.19+322056.4, which is one of the galaxies in our catalog. The quality of this model is evidenced by the residuals, highlighting the precision of the fit. For those interested in the implementation details, the code used for this analysis is provided in Appendix A.

4.2. Other available profiles

In this section, we briefly discuss the built-in profiles other than Sérsic. All these profiles but the Flat sky profile have circular radius $r$ that follows Eq. (2) to Eq. (4).

4.2.1. Exponential disk profile

In an exponential disk profile, the intensity is defined as follows:

$$I(r) = I_0 \exp \left(-\frac{r}{r_e}\right),$$

where $I_0$ is the brightness of the profile’s surface in the center (at radial distance $r = 0$), and $r_e$ is the disk scale-length.

The relation among the magnitude ($m$, $m_0$), $r_e$ and $I_0$ is given by

$$I_0 = \frac{r_{box}10^{-0.4(m_0-m)}}{2\pi q r_e^2},$$

where $q$ is the axis ratio, and $r_{box}$ is as defined in Eq. (6).

4.2.2. Modified Ferrer profile

The modified Ferrer profile, which is characterized by a nearly flat core and a rapidly truncated shape on the periphery, was originally proposed by Ferrers (1877) and later modified by Laurikainen et al. (2005). The intensity is defined as follows:

$$I(r) = I_0 \left[1 - \left(\frac{r}{r_{out}}\right)^{2-b}\right]^a,$$

where $I_0$ is also the central surface brightness parameter, $r_{out}$ is the outer truncation radius, and $b$ and $a$ are parameters governing the slopes of the truncation and core, respectively.

The relation between the magnitude ($m$, $m_0$) and the profile model parameters is given by

$$I_0 = \frac{r_{box}10^{-0.4(m_0-m)}}{\pi q r_e^2 a b \left(1 + \frac{2}{b}\right)},$$

where $q$ is also the axis ratio and $\beta$ is the beta function defined in Eq. (6). The modified Ferrer profile is particularly suitable to model the bar structure in galaxies (Blázquez-Calero et al., 2020; Dalla Bontà et al., 2018).

4.2.3. Empirical (modified) king profile

Since its initial illustration by King (1962), the empirical King profile has been extensively used to fit both galactic and extragalactic globular clusters (BAOlab2 presented by Larsen, 2014; Chies-Santos
Fig. 2. G-band galmoss fitting result of J162123.19+322056.4, showing a good fitting quality. The left four images show the model, model residuals (data-model), data, and the residual distribution. On the right, a one-dimensional projection is shown, where the upper panel represents the model distribution with a black line and the orange filled markers with error bars represent the galaxy image’s flux distribution and its uncertainty. The lower panel represents the flux residual with blue filled markers with error bars.

et al., 2007; Bonatto and Bica, 2008; Tripathi et al., 2023) The intensity of this profile is defined as follows:

$$ I(r) = I_0 \left[ 1 - \frac{1}{\left(1 + \left(\frac{r}{r_c}\right)^{1/\alpha}\right)^{1/\alpha}} \right]^n $$

(11)

Here, $I_0$ is also the central surface brightness parameter. The core radius, $r_c$, signifies the scale at which the density starts to deviate from uniformity, while $r_t$, the truncation radius, marks the boundary of the cluster. The global power-law factor, $\alpha$, dictates the rate at which the density declines with distance from the center. The concentration of stars in globular clusters can be defined using the parameters $r_t$ and $r_c$, denoted by $c = \log \left(\frac{r_t}{r_c}\right)$.

4.2.4. Gaussian profile

In a Gaussian profile, the intensity is defined as follows:

$$ I(r) = I_0 \exp \left( - \frac{r^2}{2\sigma^2} \right) $$

(12)

where $I_0$ is also the central surface brightness parameter, and $\sigma$ is the radial dispersion. In the galmoss implementation, the Full Width at Half Maximum (FWHM = 2.354$\sigma$) is used instead. The classical application of the Gaussian profile includes modeling a simple PSF and point sources.

4.2.5. Moffat profile

The Moffat profile (Moffat, 1969), commonly used for modeling a realistic telescope PSF, defines its intensity as follows:

$$ I(r) = I_0 \left[ 1 + \left(\frac{r}{r_d}\right)^2 \right]^{-n} $$

(13)

where

$$ r_d = \frac{\text{FWHM}}{2\sqrt{2} - 1} $$

and $I_0$ also is the central density. The concentration index, $n$, dictates whether the distribution is more Lorentzian-like ($n=1$) or Gaussian-like ($n \to \infty$).

When considering the ellipse with axis ratio $q$ and boxiness parameter ($r_{\text{box}}$ in Eq. (6)), the relationship between the observed magnitude $m$ and profile model parameter is

$$ I_0 = \frac{r_{\text{box}}(n-1)(n-2)(d-m_0)}{\pi q r_d^2} $$

(15)

Similar to the Gaussian profile, the Moffat profile can also be used to model point sources.

4.2.6. Flat sky

Unlike other radial light profiles, in galmoss, the sky profile is controlled by the sky mean value $I_{\text{sky}}$ across all pixels without a radial matrix:

$$ I = I_{\text{sky}} $$

(16)

In galmoss, the only parameter needed in sky Profile is $I_{\text{sky}}$. If a sky profile varies with position (radius), it can be included as a user-defined profile.

5. Image fitting and evaluation

Following the generation of model images, galmoss implements an extended $\chi^2$ likelihood and gradient descent optimization, leveraging python and pytorch functionalities. In addition, uncertainty estimation is available.

5.1. Maximum likelihood estimation

The galmoss package employs a $\chi^2$ likelihood:

$$ \ln L = -\frac{k}{2} \ln 2 - \ln \Gamma(k/2) + \left(\frac{k}{2} - 1\right) \ln x^2 - \frac{x^2}{2} $$

(17)
Fig. 3. Panels comparing the measured Sérsic profile parameters for the selected ∼ 8,000 SDSS galaxies, along with $R^2$ estimation. The panels each display galaxy magnitude ($m$), position angle ($\theta$), axis ratio ($q$), effective radius ($r_e$) and Sérsic index ($n$). The transparency of the dots illustrates their concentration. The blue line shows where galfit result equals galmoss result. The histogram on top and right illustrates the distribution of measured values for galmoss and galfit, respectively. The comparison shows proper consistency in all panels.

$$\chi^2 = \sum_{i=1}^{N_i} \sum_{j=1}^{N_j} \frac{(D_{ij} - M_{ij})^2}{\sigma_{ij}^2},$$

where $M_{ij}$ represents the model at pixel $i,j$, and $\sigma_{ij}$ represents the uncertainty of $D_{ij}$. In the $\chi^2$ distribution, the degrees of freedom, $k$, is the difference between the number of pixels with galaxy flux that are not included in the mask and the number of model fit parameters.

Internally galmoss adopts gradient descent for parallel fitting. Though slower in convergence than other traditional methods (e.g. Levenberg–Marquardt) it has lower computational and memory demands, enabling efficient parallel fitting on GPUs.

5.2. Confidence intervals

Galmoss employs two strategies to estimate uncertainties in galaxy images, primarily using a covariance matrix based on Gaussian distribution assumptions (Hogg et al., 2010). The parameter uncertainties are calculated from the covariance matrix’s diagonal, derived using the Jacobian matrix (e.g., Gavin, 2022) for computational efficiency. This method approximates uncertainties as $\sigma_p = \sqrt{\text{diag}[J^TWJ]}$, where $J$ is the Jacobian matrix, and $W$ is diagonal with $W_{ij} = 1/\sigma_{ij}^2$, offering a 1-$\sigma$ confidence level for the parameters. The second methodology for uncertainty estimation is bootstrapping, which involves resampling with replacement each galaxy images multiple times (typically 100 iterations) and refitting them. The uncertainty is given by $\sigma^2 = \frac{1}{M} \sum_{i=1}^{M} [m_i - \bar{m}]^2$, where $\bar{m}$ is the mean value of the estimated parameter. Fig. 4 illustrates a comparison of uncertainty estimation methods for a random subset of galaxies. We display the uncertainties determined through bootstrap analysis (red error bars) and those derived from the covariance matrix (blue error bars). Generally, bootstrap uncertainties encompass a broader range, suggesting a more conservative estimate.
6. Results

To evaluate the performance of galMoss against established methods, we performed a validation test by fitting single galaxy profiles from the same dataset and comparing the structural parameters, such as the Sérsic index and effective radius, with those obtained using galfit, as cataloged in the MPP-VAC-DR17. This comparison acts as a fundamental ‘sanity check’ to verify the reliability of galMoss’s fitting results. Fig. 3 shows overall consistency across key galaxy parameters—magnitude (m), position angle (θ), axis ratio (q), effective radius (r_e), and Sérsic index (n)—as quantified by the R^2 coefficient of determination.

The panels for magnitude, position angle, and axis ratio show a strong linear correlation, indicative of a high degree of alignment with galfit results. Specifically, in the position angle panel, objects that deviate significantly from the identity line are those fitted with large axis ratios (q ≈ 1). This suggests that, in these cases, the light distributions are relatively insensitive to variations in the position angle (θ).

The parameters r_e and n, known to be challenging to fit accurately (Trujillo et al., 2001), demonstrate a linear relationship, albeit with greater dispersion compared to other parameters. Notably, in the Sérsic index panel, galMoss values are generally lower than those derived from galfit at higher values of n. This discrepancy could be attributed to variations in image quality and the characteristics of the optimization algorithms used in PyTorch. For an in-depth analysis of this observed bias, we direct the reader to Appendix E.

In addition to the aforementioned test, we conducted an independent speed comparison using galfit and galMoss using the same dataset and initial values. The fitting of 8,289 galaxies with galMoss completed in roughly 10 minutes—six times faster than with serial galfit. Fig. 5 demonstrates this efficiency gain across different batch sizes, showing galMoss’ speed advantage becoming more pronounced as batch size increases, up to the limits of GPU capacity.

7. Conclusions

In this study, we introduce galMoss, an open-source software package specifically designed for fitting galaxy light profiles on large datasets, ideally suited for the LSST era. Built on the torch framework, galMoss provides efficient 2D surface brightness profile fitting for batches of galaxy images, with the added benefit of GPU acceleration. It features a user-friendly interface that allows for the easy definition of parameters and their ranges. Moreover, galMoss is capable of efficiently quantifying uncertainty through both covariance matrix and bootstrap methods. The package also supports the integration of new profiles, making it an adaptable and versatile tool for statistical analysis in galaxy structure studies.

Benchmark tests reveal that galMoss can achieve speedup gains of up to 6 times compared to galfit, depending on the sample size. One novel aspect of galMoss is its use of native parallelization to process multiple observational fields of single galaxies as elements of a single multidimensional tensor, thereby enhancing its speed through efficient PyTorch GPU-accelerated matrix calculations and memory usage (Paszke et al., 2019). This approach contrasts with astrophot, which focuses on larger fields with multiple galaxies or joint multi-band fitting. A current limitation of galMoss is the requirement to manually select profile models and initial parameters, along with its sub-optimal GPU utilization for small galaxy batches—areas for improvement in future versions.

galMoss is freely available at GitHub,3 Zenodo4 and listed in the Python Package Index.5 The readthedocs6 contains example usages, along with an overview of the package.

CRediT authorship contribution statement

Mi Chen: Writing – original draft, Software. Rafael S. de Souza: Writing – review & editing, Supervision. Quanfeng Xu: Writing – review & editing. Shiyin Shen: Writing – review & editing, Supervision. Ana L. Chies-Santos: Writing – review & editing. Renhao Ye: Writing – review & editing. Marco A. Canossa-Gosteiński: Writing – review & editing. Yanping Cong: Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

We thank Gongyu Chen for helping with the logic and language flow of the paper. We thank MengTing Shen for helping with the preparation of SDSS dataset. We thank Zhu Chen and Ziqi Ma for their help in learning the usage of sextractor. We thank QiQi Wu and Shuairu Zhu for their help in galMoss application.

ACS acknowledges funding from the brazilian agencies Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and the Rio Grande do Sul Research Foundation (FAPEG/S) through grants CNPq-11153/2018-6 and FAPERGS/CAPES 19/2551-0000696-9. SS thanks research grants from the China Manned Space Project with NO. CMS-CSST-2021-A07, the National Key R&D Program of China (No. 2022YFF0503402, 2019YFA0405501), National Natural Science Foundation of China (No. 12073059 & 12141302) and Shanghai Academia/Technology Research Leader, China (22XD1404200). Funding for the Sloan Digital Sky Survey IV has been provided by the Alfred P. Sloan Foundation, the U.S. Department of Energy Office of Science, and the Participating Institutions. SDSS acknowledges support and resources from the Center for High-Performance Computing at the University of Utah. The SDSS web site is www.sdss4.org.

3 https://github.com/Chenmi0619/GALMos
4 https://doi.org/10.5281/zenodo.10654784
5 https://pypi.org/project/galmoss/
Appendix A. How to use galmoss: single Sérsic case

Here, we demonstrate how to fit a single Sérsic profile to SDSS image data using the galmoss package.

First, we need to load the necessary packages.

```python
import Galmoss as gm
```

Next, we need to define the parameter objects and associate them with profile instances. The initial estimates of the galaxy parameters are provided by sextractor. Notably, we do not include the boxiness parameter in this simple example, despite its availability within the galmoss framework.

```python
# define parameter objects and profile
ser = gm.lp.Sersic(
    cen_x=gm.p(65.43),
    cen_y=gm.p(64.95),
    pa=gm.p(-81.06, angle=True),
    eff_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)
)
```

The comprehensive dataset object can be formulated utilizing the image sets (galaxy image, sigma image, PSF image, mask image) together with the chosen profiles.

```python
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",
    index=2,
    result_path="/test_repo",
)
```

After initializing the hyperparameter during the fitting process, fitting could start. Subsequently, we run the uncertainty estimation process.

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

When the fitting process is completed, the fitted results and the img_blocks are saved in corresponding path.

Appendix B. How to use galmoss: bulge+disk case

Here, we demonstrate how to use a combination of two Sérsic profiles to make disk and bulge decomposition on SDSS image data using the galmoss package.

```python
import Galmoss as gm
```

Upon importing the package, the subsequent step entails defining parameter objects. To ensure that the center parameter within both profiles remains the same, it suffices to specify the center parameter once and subsequently incorporate it into various profiles.

```python
xcen = gm.p(65.97)
ycen = gm.p(65.30)
```

For a quick start, we let the disk and bulge profile share the initial value from the sextractor, with an initial Sérsic index of 1 for the bulge component and 4 for the disk component.

```python
disk = gm.lp.Sersic(
    cen_x=xcen,
    cen_y=yacen,
    pa=gm.p(58.70, angle=True),
    axis_r=gm.p(0.75),
    eff_r=gm.p(4.09, pix_scale=0.396),
    ser_n=gm.p(4),
    mag=gm.p(17.97, M0=22.5)
)
bulge = gm.lp.Sersic(
    cen_x=xcen,
    cen_y=yacen,
    pa=gm.p(55., angle=True),
    axis_r=gm.p(0.76),
    eff_r=gm.p(4.09, pix_scale=0.396),
    ser_n=gm.p(11),
    mag=gm.p(17.97, M0=22.5)
)
```

Compared to the single profile case, we only need to change the code snippet of profile definition. We choose to use bootstrap to calculate the uncertainty here.

```python
dataset = gm.Data_Box(  
galaxy_index="J100247.00+042559.8"),  
image_path="/J100247.00+042559.8_image.fits",  
sigma_path="/J100247.00+042559.8_sigma.fits",  
psf_path="/J100247.00+042559.8_psf.fits",
mask_path="/J100247.00+042559.8_mask.fits",
image_block_path="/test_repo",
result_path="/test_repo")
```

Fig. B.6 show the decomposition result, with the residual demonstrating a high quality of fitting.

Appendix C. Example code for definition of a set of galaxy profiles

Here, we provide an example that shows how to define a combination of profiles and extract the model image from the image function for an initial review. Fig. C.7 shows a model image featuring a central galaxy with a disk, bulge, and bar, each defined by two Sérsic profiles and a Ferrer profile, respectively. Additionally, there is a side galaxy defined by a Sérsic profile, two point sources defined by Moffat profiles, and an open cluster defined by a King profile. All of these are convolved with a PSF image produced by a Gaussian profile.

```python
import numpy as np
import Galmoss as gm
import torch.optim as optim

upon importing the package, the first sub-model represents the central galaxy, comprising a disk, bulge, and bar.

```python
# The galaxy center
xcen = gm.p(65.)
ycen = gm.p(64.)
```

```python
# Define parameters and profiles
Disk = gm.lp.Sersic(
    cen_x=xcen,
    cen_y=ycen,
    pa=gm.p(55., angle=True),
    axis_r=gm.p(0.44),
    axis_r=gm.p(0.44),
    eff_r=gm.p(4.09, pix_scale=0.396),
    ser_n=gm.p(11),
    mag=gm.p(17.97, M0=22.5),
    box=gm.p(0.1)
)
```

```python
Bulge = gm.lp.Sersic(
    cen_x=xcen,
    cen_y=ycen,
    pa=gm.p(0., angle=True),
```
Fig. B.6. G-band galmoss decomposition result of J100247.00+042559.8, showing a good fitting quality. The left six images show the total model, disk model, bulge model, data, model residuals (data-model), and the residual distribution. On the right, a one-dimensional projection is shown. The upper panel represents the model distribution with a black line, and the orange dots with error bars represent the galaxy image's flux distribution and its uncertainty. The red dashed line and blue dotted line represent the distribution of the bulge component and the disk component, respectively. The lower panel represents the flux residual with blue dots with error bars.

The second sub-model is a side galaxy.

```python
# The idealized PSF
PSF = gm.lp.Gaussian(cen_x=gm.p(20), cen_y=gm.p(20), axis_r=gm.p(0.2), inten=gm.p(1/(2*np.pi*((0.6/0.396) ** 2))), fwhm=gm.p(1.2, pix_scale =0.396),)
```

The third sub-models are two point sources. They share the dispersion $\sigma = 1.515''$ with the idealized PSF image, both of which are produced by a Gaussian profile.

```python
# Two point-souce
P1 = gm.lp.Gaussian(cen_x=gm.p(75), cen_y=gm.p(80), axis_r=gm.p(1), inten=gm.p(0.1), fwhm=gm.p(1.2, pix_scale =0.396),)
```

```python
# Two point-souce
P2 = gm.lp.Gaussian(cen_x=gm.p(88.), cen_y=gm.p(40.), axis_r=gm.p(1), inten=gm.p(0.1), fwhm=gm.p(1.2, pix_scale =0.396),)
```

```python
# The second sub-model is a side galaxy.
Sersic_s = gm.lp.Sersic(cen_x=gm.p(45.43), cen_y=gm.p(100.95), axis_r=gm.p(0.2), eff_r=gm.p(13, pix_scale =0.396), ser_n=gm.p(1.6), mag=gm.p(17.5, M0=22.5), box=gm.p(-0.6))
```

axis_r=gm.p(1),
eff_r=gm.p(6, pix_scale =0.396),
ser_n=gm.p(6),
mag=gm.p(21.5, M0=22.5))
```

The second sub-model is a side galaxy.

```python
Bar = gm.lp.King(cen_x=xcen, cen_y=ycen, pa=gm.p(-40., angle=True), axis_r=gm.p(0.2), mag=gm.p(21.5, M0=22.5), trunc_r=gm.p(5, pixScale =0.396), trunc_a=gm.p(0.5), trunc_b=gm.p(1.9), box=gm.p(0.1))
```

```python
axis_r=gm.p(1),
eff_r=gm.p(6, pix_scale =0.396),
ser_n=gm.p(6),
mag=gm.p(21.5, M0=22.5))
```
The total model image without convolution is the sum of three sub-components (see Fig. C.7(a)). To extract the model images, it is necessary to first define a grid, as grid generation occurs automatically only during the fit process within the fitting object.

To mimic the effect of seeing, an idealized PSF image will be generated from a relatively small grid and then convolved with the model images (except for the point sources). The result is shown in Fig. C.7(b).

Appendix D. User-defined profile

In Section 4.2.6, we introduced the sky profile. Here, we show how to integrate a more realistic sky profile as an example of a user-defined profile. In this new sky profile, the sky intensity is defined as follows:

\[ I = I_0 + k_x (x - x_0) + k_y (y - y_0). \]  

where \((x_0, y_0)\) is the geometric center of the image, \(I_0\) is the sky value at \((x_0, y_0)\), and \((k_x, k_y)\) describes the variation of the sky value along the \(x\) and \(y\) axes.

To integrate this new profile, we need to define a new class, which we have named *NewSky*. This class should include a function that guides the generation of the image model from the given parameters. Here are some caveats. Firstly, every profile class should inherit from the basic class *LightProfile* to access general light profile functions. Next, the parameters should be loaded into the profile class in the _`init_` function, and set as attributions (e.g., self.sky_0 = sky_0).

The equation for the profile is defined within the `image_via_grid_from` function. Parameter values are extracted after the mode value, which has a default value of `updating_model`. This mode calls for values that are continuously updated throughout the fitting process and have already been broadcast to a suitable shape for multi-dimensional matrix calculations.

A newly defined profile can be used as follows:

### Appendix E. The fitted bias in high Sérsic index end

As shown in Fig. 3, we observe a general bias in the high Sérsic index \(n\) in the comparison results. Galaxies with large \(n\) values (e.g., \(n > 6\)) in the galfit results mostly exhibit lower values (e.g., \(n < 6\)) in the galmoss results.

To further investigate the type of high \(n\) galaxies, we use the morphological classification afforded by catalog MDLM-VAC-DR17. We identify 8,289 galaxies into 2,849 ETGs and 5,440 LTGs, and further identify ETGs into 1,891 Elliptical galaxies, 793 S0 galaxies and 165 undefined galaxies. We find that ETGs account for 34.37% of the total galaxy samples and represent 88.78% in the biased region. To investigate further, we plot a histogram for the \(n\) value in E and S0 (Fig. E.8). In the plot, the histogram almost coincides between galmoss fitted results and galfit fitted results in 50 galaxies. However, the distribution of the histogram is very different between the galmoss fitted results and the galfit fitted results in elliptical galaxies. This result reveals that Elliptical galaxies contribute the most to the inconsistency in fitted \(n\) values.

There are several reasons for the observed result. One of them is the limitation of observation. Most elliptical galaxies have a steep central core and a flat outer wing. Due to the poor signal-to-noise ratio in the flat wing, the \(x^2\) is insensitive to the change of \(n\), which acts as a small gradient during the optimization. Another possible reason is related to the properties of the optimizer. Most prevalent optimizers in pytorch use self-adaptive learning rates which reduce the learning rate under low gradient conditions to fasten convergence. As a result, the sensitivity of \(n\) is further reduced at the high-value end compared to galfit, which adds to the difference in results between the two software in this case.
Fig. E.8. The histogram shows the distribution of $a$ values in E and S0 galaxies. Solid lines represent $	ext{galmoss}$ results, and dashed lines are for galfit. Purple and pink correspond to S0 and E, respectively. It is evident from the inconsistency between the values fitted by galmoss and galfit that Elliptical galaxies contribute most to the inconsistency in the high end of $a$ fitted results.

References


