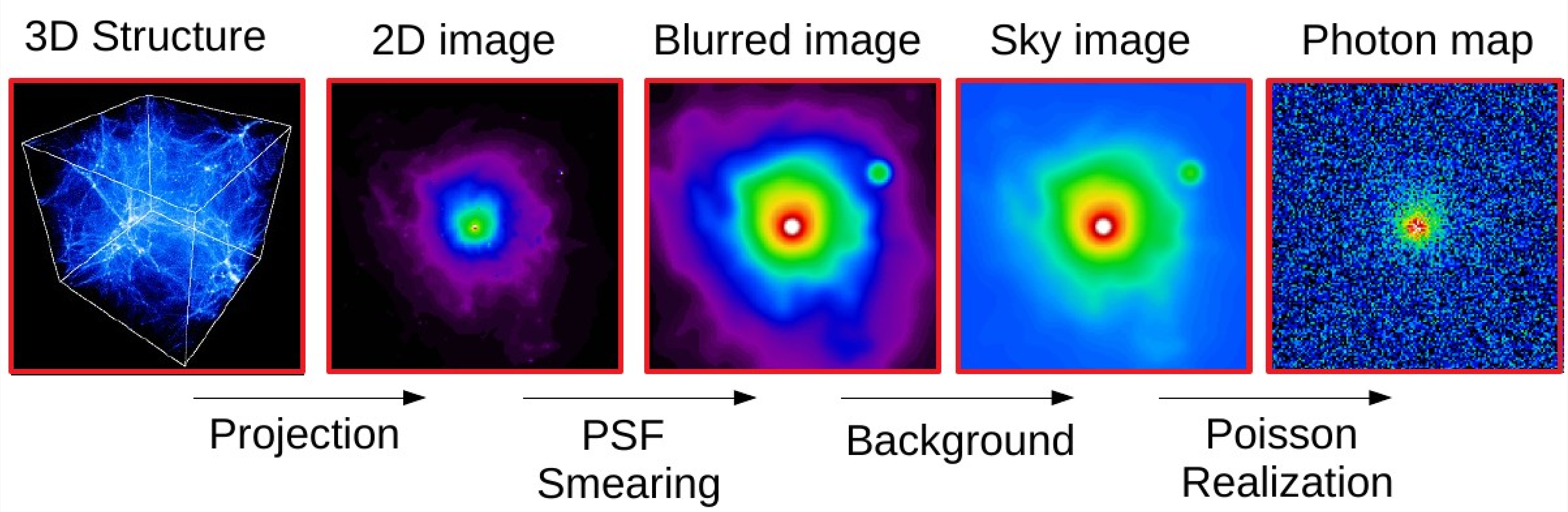
Bayesian analysis of galaxy cluster properties with *pyproffit* and *hydromass*

Dominique Eckert, Stefano Ettori

Deconvolution amplifies noise

- Observed galaxy cluster profiles are noisy realizations of *projected* and *PSF-convolved* physical quantities

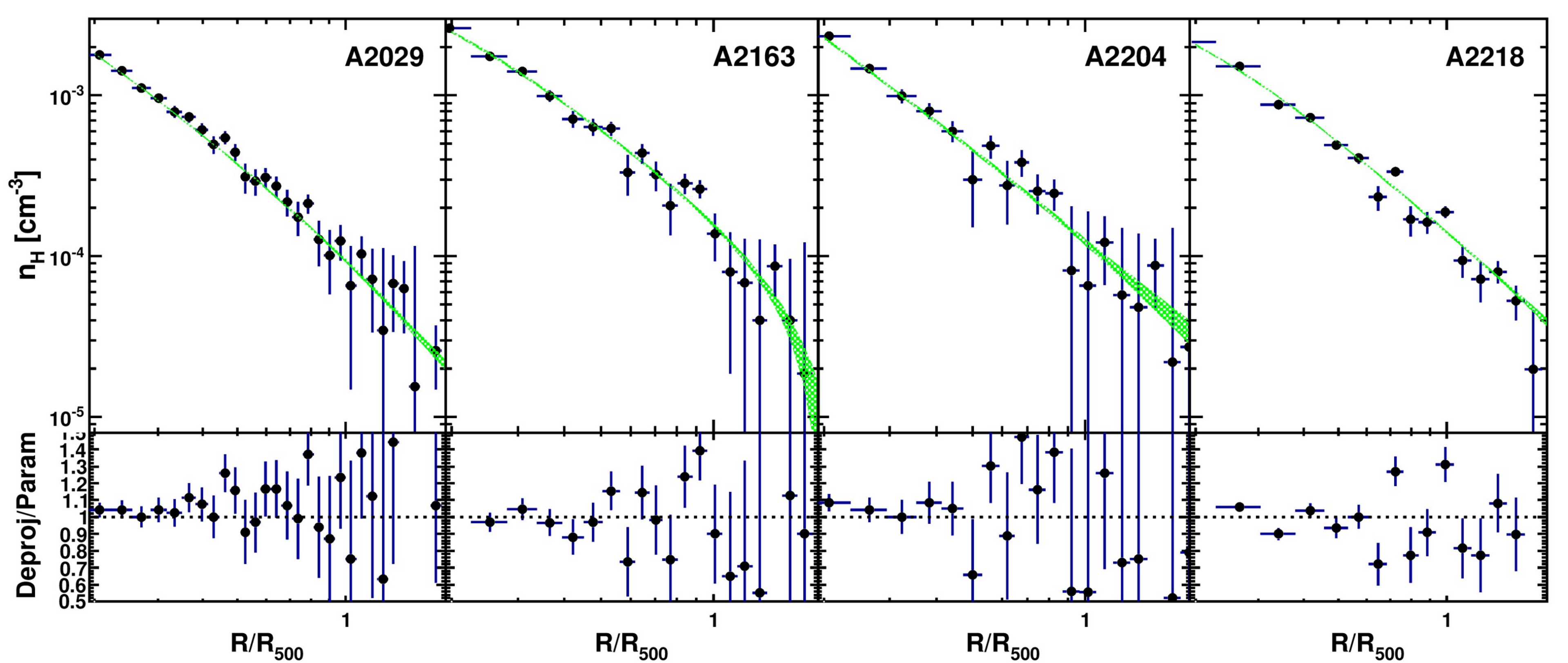


- The convolution kernel smoothes fluctuations, thus deconvolution has the opposite effect

“Traditional” approaches

Parametric form: Only as good as what the adopted function can reproduce

Direct inversion: Amplifies noise, depends on the chosen binning, can lead to unphysical solutions



Eckert et al. 2013

Decomposition on a basis of functions

- **Multiscale approach**: decompose the observed profile onto a basis of functions which can be individually deprojected/deconvolved

$$\epsilon_{3D}(r) = \sum N_i \Phi_i(r)$$

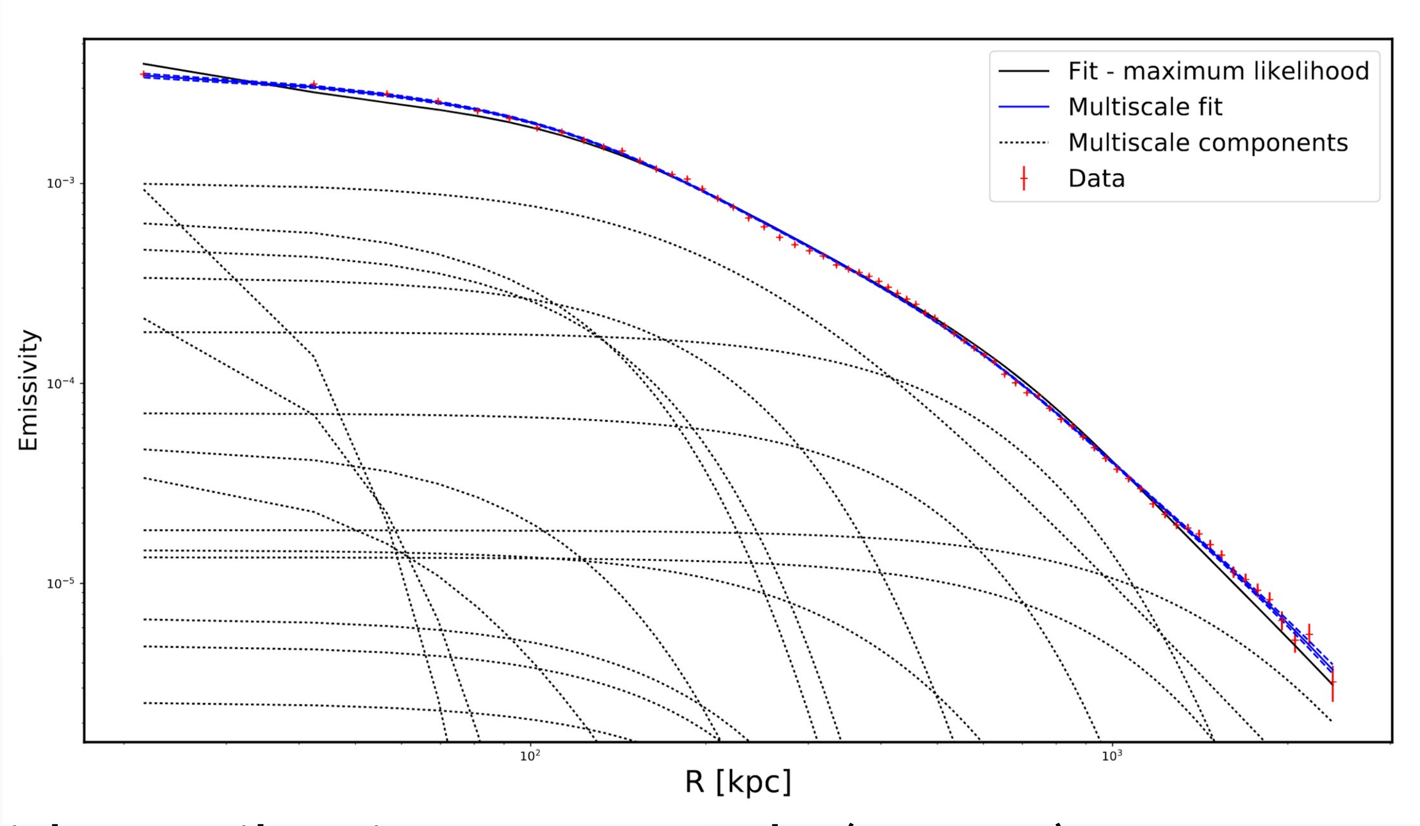
$$S_X(\omega) = \int_{-\infty}^{\infty} \epsilon_{3D}(r)\, dz \qquad \omega = \sqrt{x^2 + y^2}$$

$$= \sum \alpha_i \phi_i(\omega)$$

The relations

$$\Phi_i \rightarrow \phi_i \qquad N_i \rightarrow \alpha_i$$

can be computed once and for all

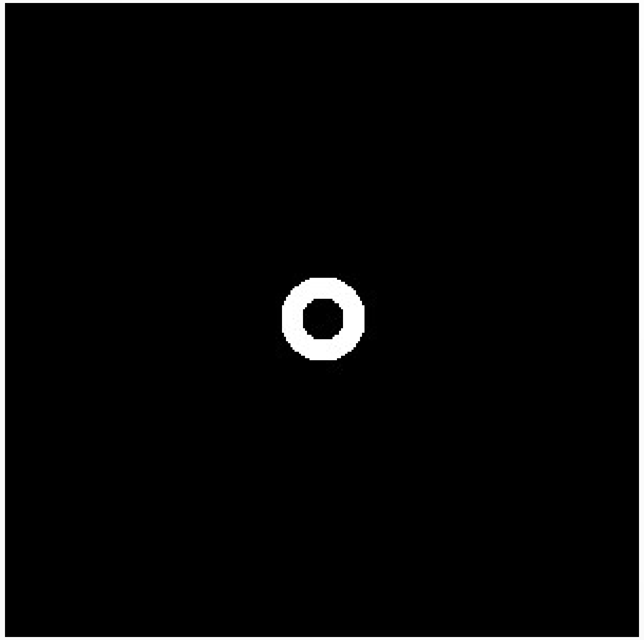


- **Optimization** performed with Hamiltonian Monte Carlo (PyMC3)

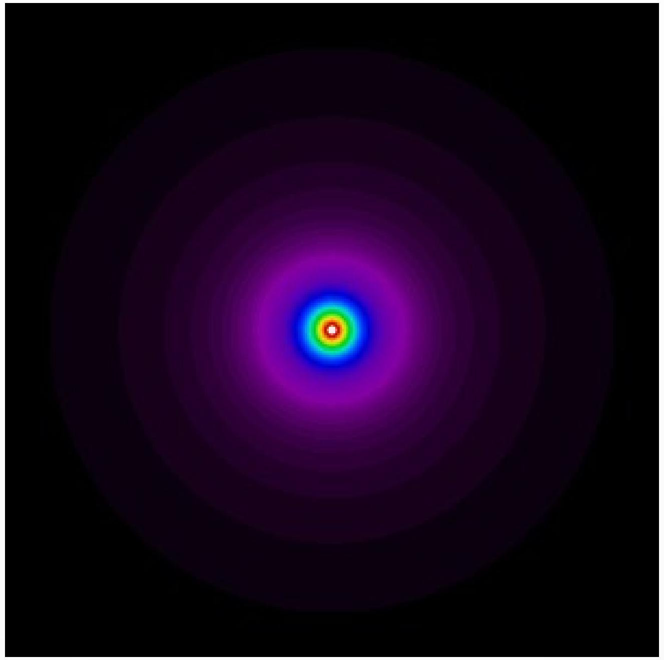
PSF deconvolution

- To account for PSF smearing we create a PSF mixing matrix using FFT

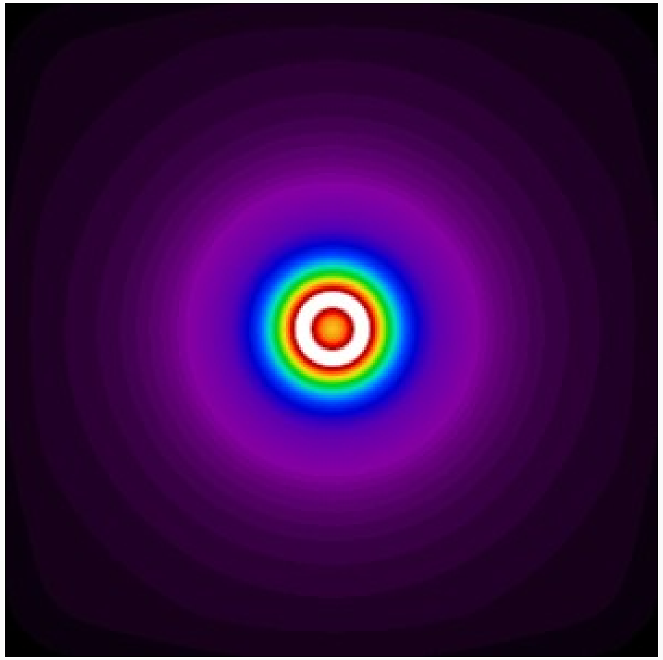
Annulus



Kernel



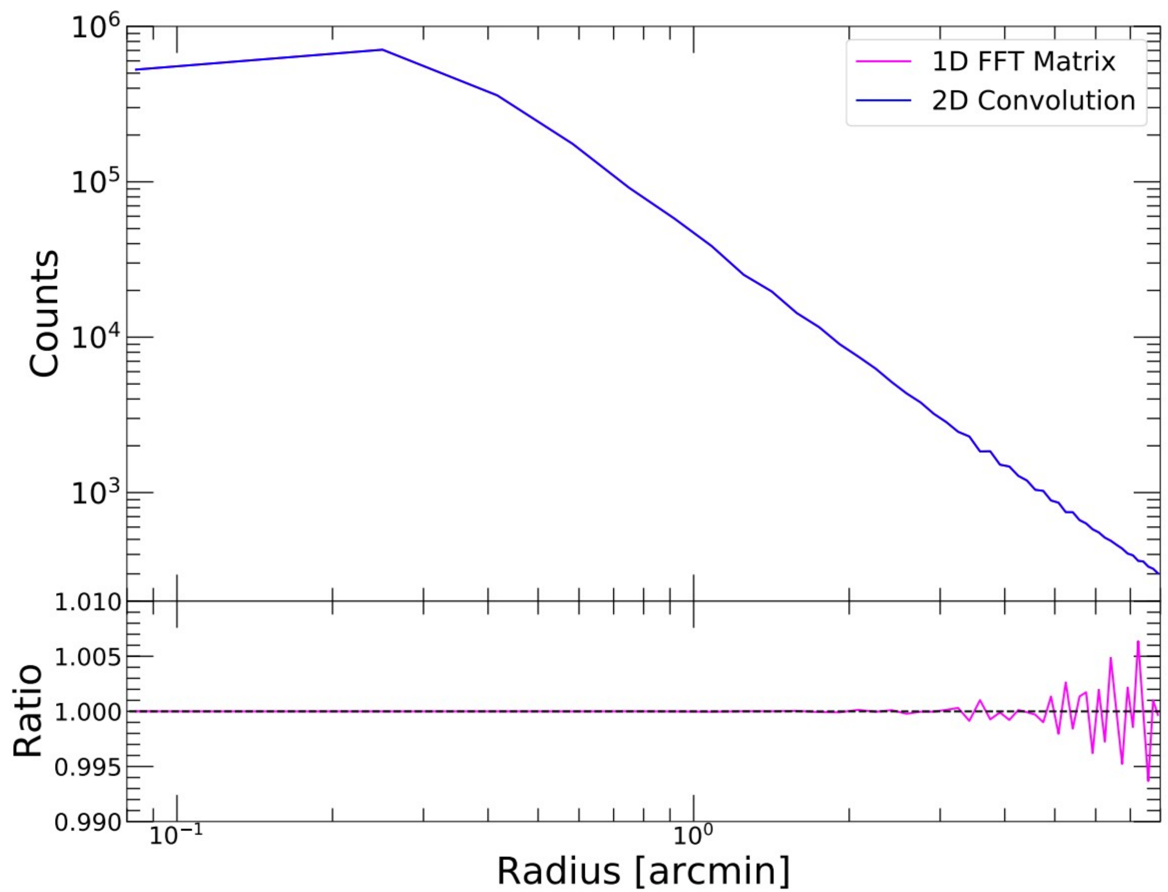
Convolved



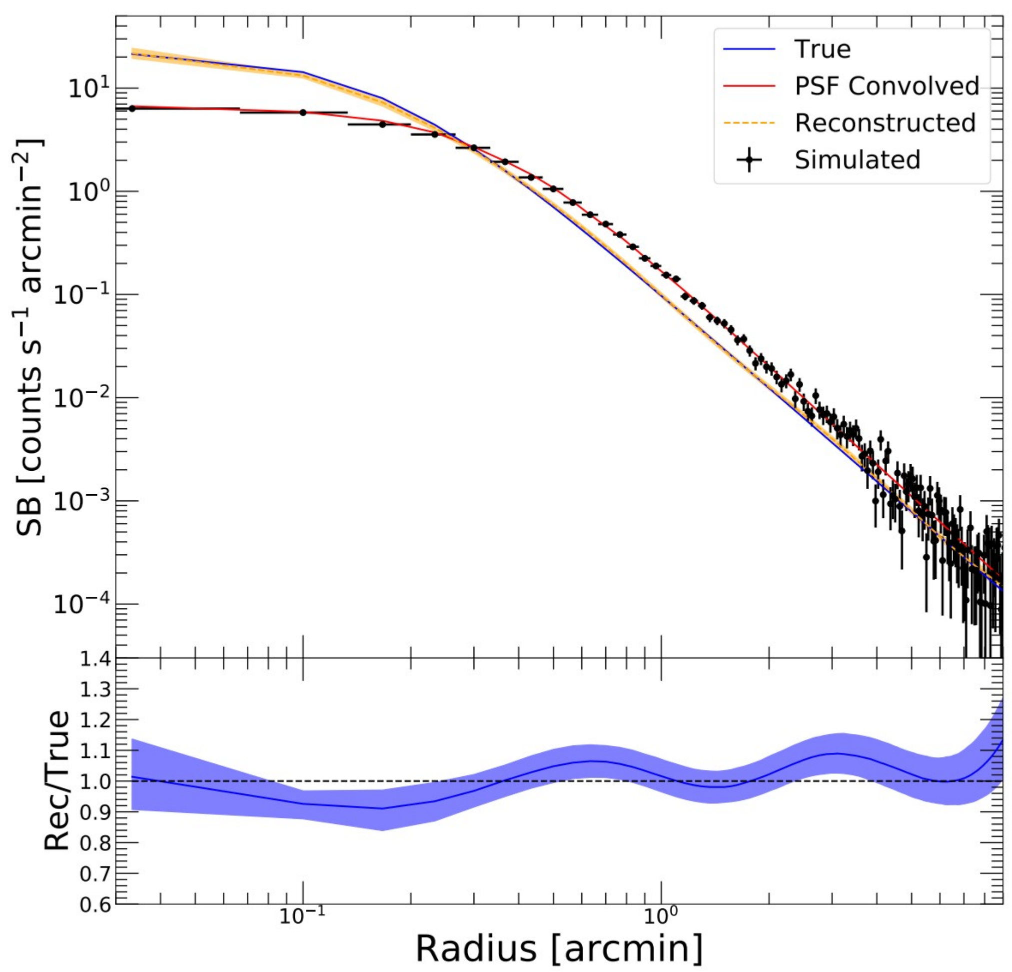
We count the fraction of photons being recorded annulus by annulus, and repeat the operation for each annulus

PSF convolution: tests

Point source



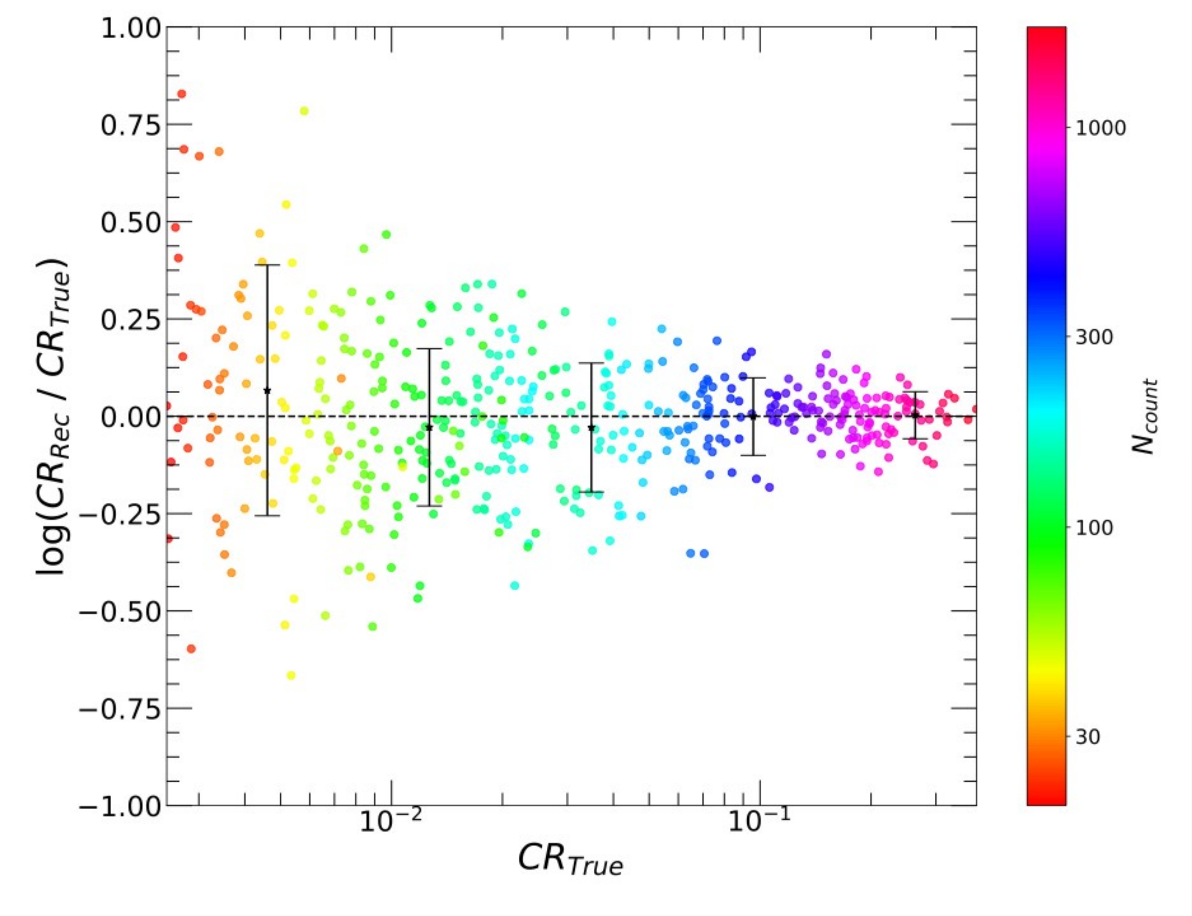
Beta model

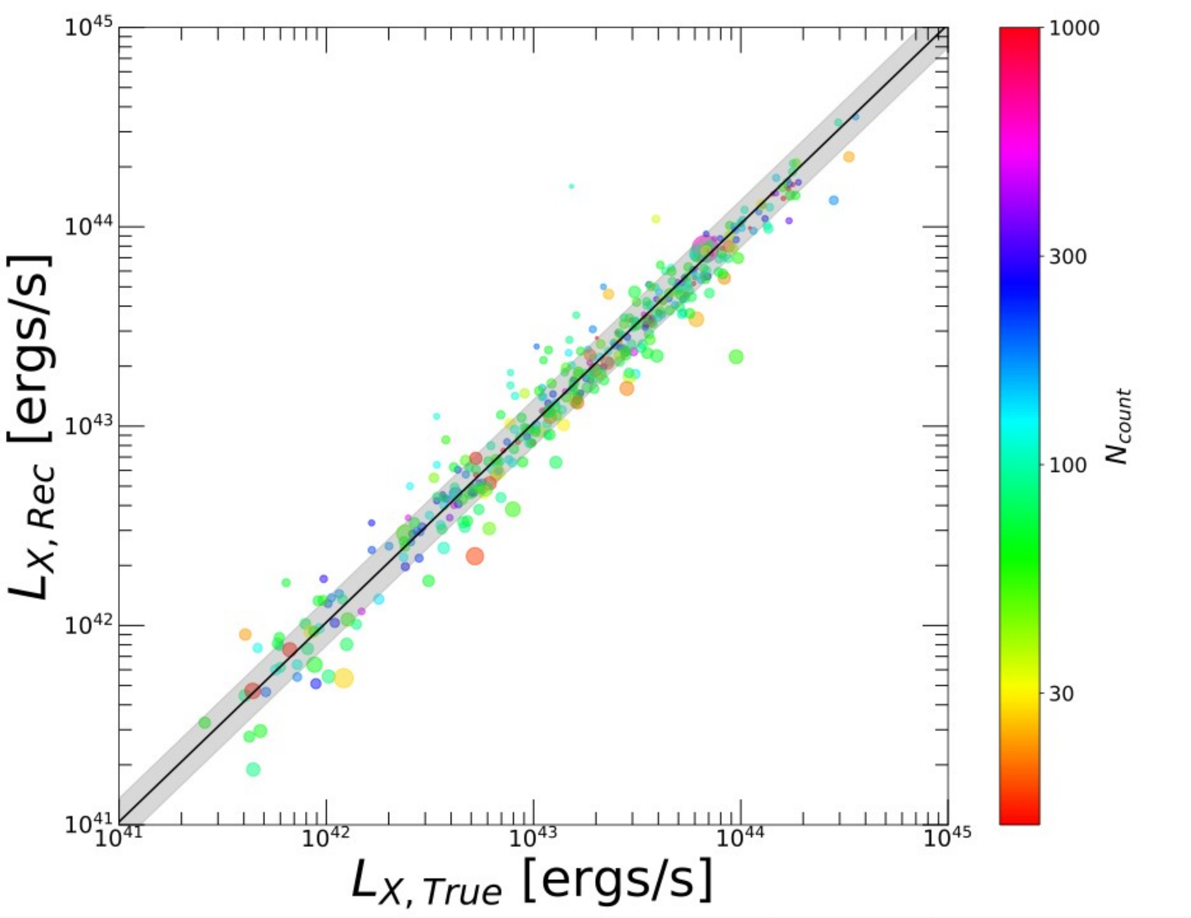


Eckert et al. 2020

Integrated quantities

- The code is able to determine accurate luminosities, including core-excised ones, even for sources detected with 30 counts

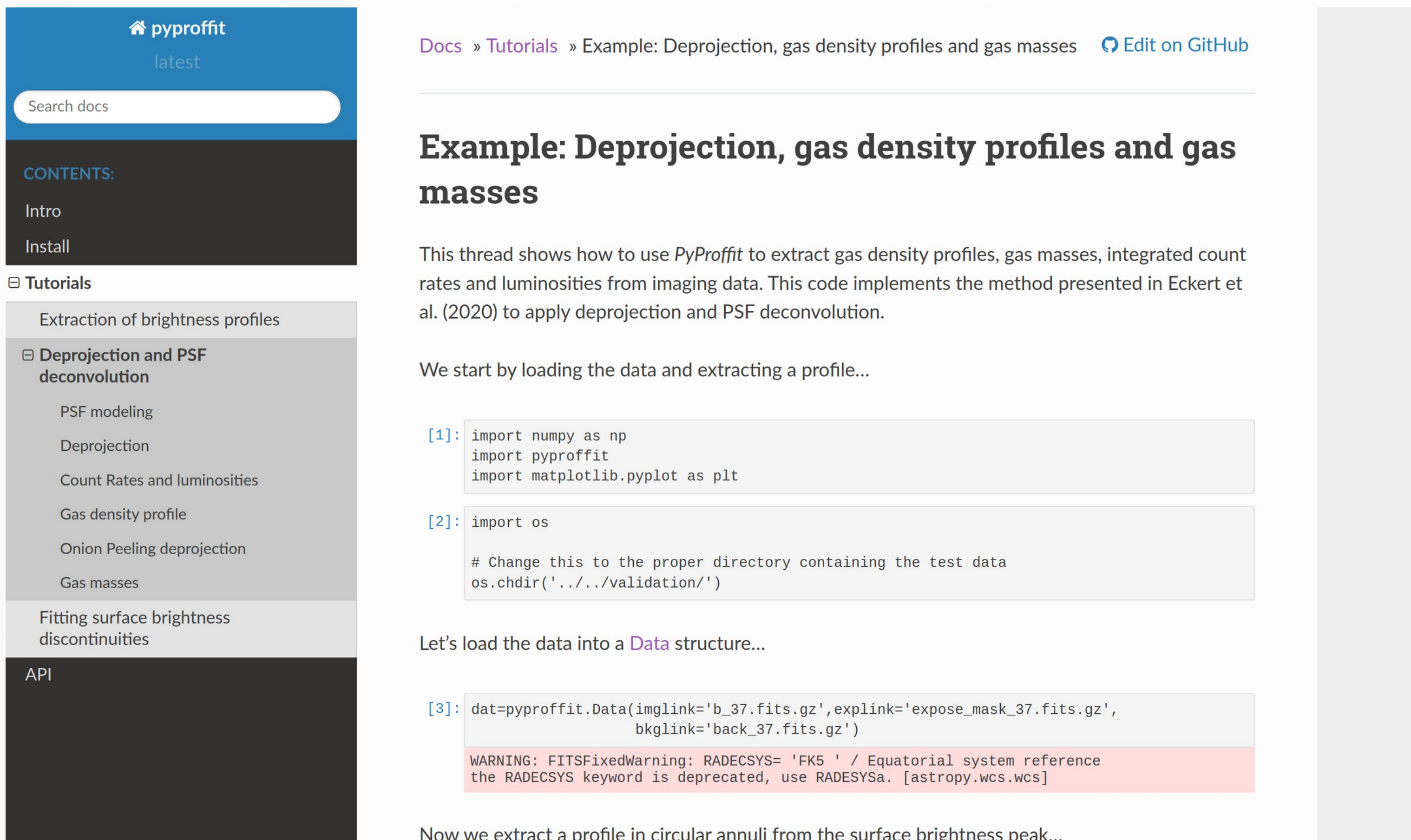




Eckert et al. 2020

Pyproffit: a Python package for surface brightness analysis

- The code is distributed in the public Python package *pyproffit*



pyproffit
latest
Search docs
CONTENTS:
Intro
Install
⊟ Tutorials
Extraction of brightness profiles
⊟ Deprojection and PSF deconvolution
PSF modeling
Deprojection
Count Rates and luminosities
Gas density profile
Onion Peeling deprojection
Gas masses
Fitting surface brightness discontinuities
API

Docs » Tutorials » Example: Deprojection, gas density profiles and gas masses   Edit on GitHub

Example: Deprojection, gas density profiles and gas masses

This thread shows how to use *PyProffit* to extract gas density profiles, gas masses, integrated count rates and luminosities from imaging data. This code implements the method presented in Eckert et al. (2020) to apply deprojection and PSF deconvolution.

We start by loading the data and extracting a profile...

```
[1]: import numpy as np
     import pyproffit
     import matplotlib.pyplot as plt
```

```
[2]: import os

     # Change this to the proper directory containing the test data
     os.chdir('../../validation/')
```

Let's load the data into a Data structure...

```
[3]: dat=pyproffit.Data(imglink='b_37.fits.gz',explink='expose_mask_37.fits.gz',
                        bkglink='back_37.fits.gz')

     WARNING: FITSFixedWarning: RADECSYS= 'FK5 ' / Equatorial system reference
     the RADECSYS keyword is deprecated, use RADESYSa. [astropy.wcs.wcs]
```

Now we extract a profile in circular annuli from the surface brightness peak...

https://pyproffit.readthedocs.io

Non-parametric log-normal mixture deprojection

- We suppose that the function of interest (temperature profile) can be described as a linear combination of a large number (P) of log-normal functions

$$T(r) = \sum_{i=1}^{P} N_i \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(\frac{-(\ln(r) - \ln(\mu_i))^2}{2\sigma_i^2} \right)$$

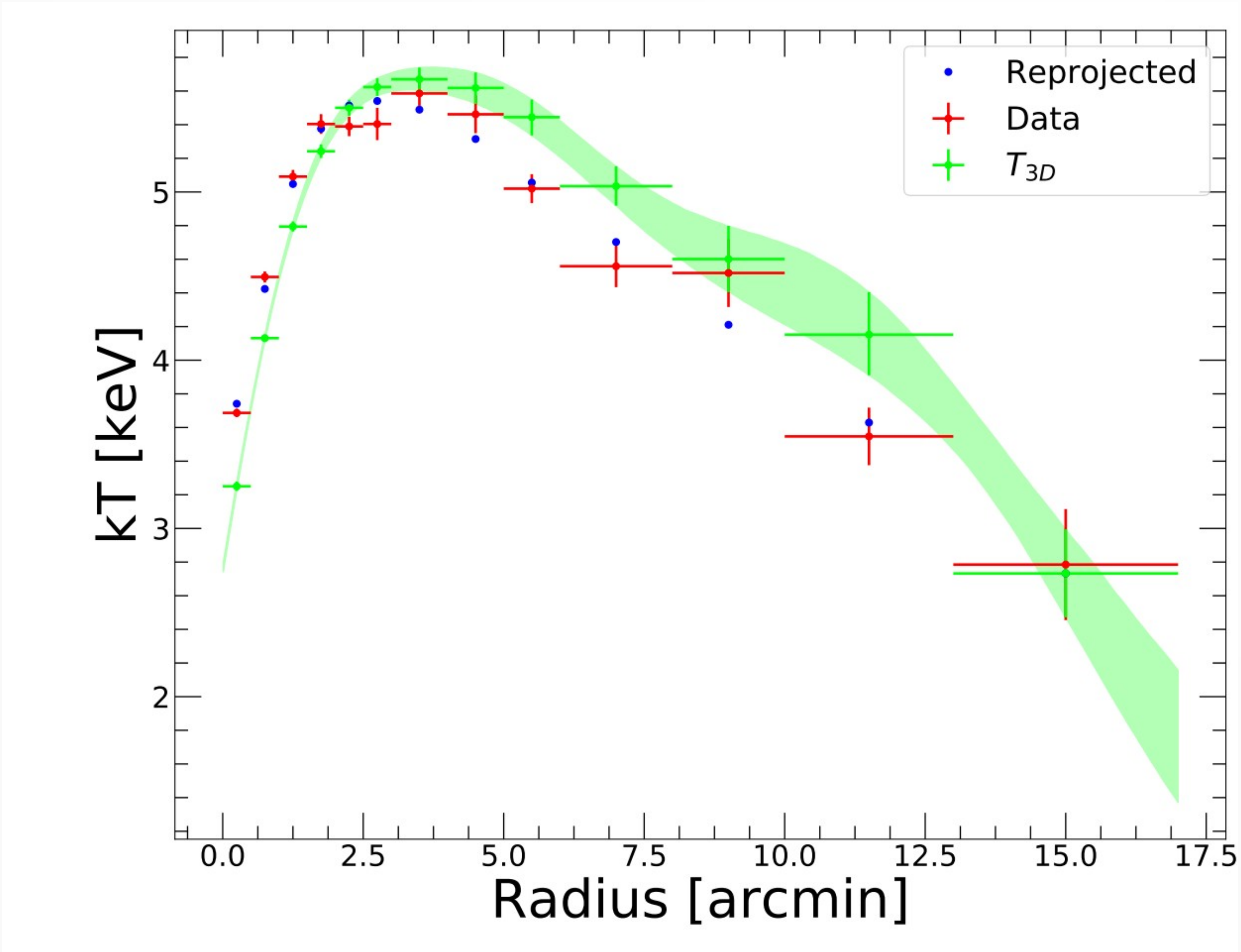
- For a basis of functions $\{G_i\}$ characterized by predefined means $\{\mu_i\}$ and standard deviations $\{\sigma_i\}$, the temperature profile can be determined by optimizing the normalizations $\{N_i\}$,

$$\log L = -0.5 \sum_{j=1}^{N} \frac{(T_j - T_{model}(r_j))^2}{\sigma_{T,j}^2}$$

with T_{model} a function of T(r) given above.

Non-parametric 3D temperature profile reconstruction

- Example: A1795



Red: 2D temperature data

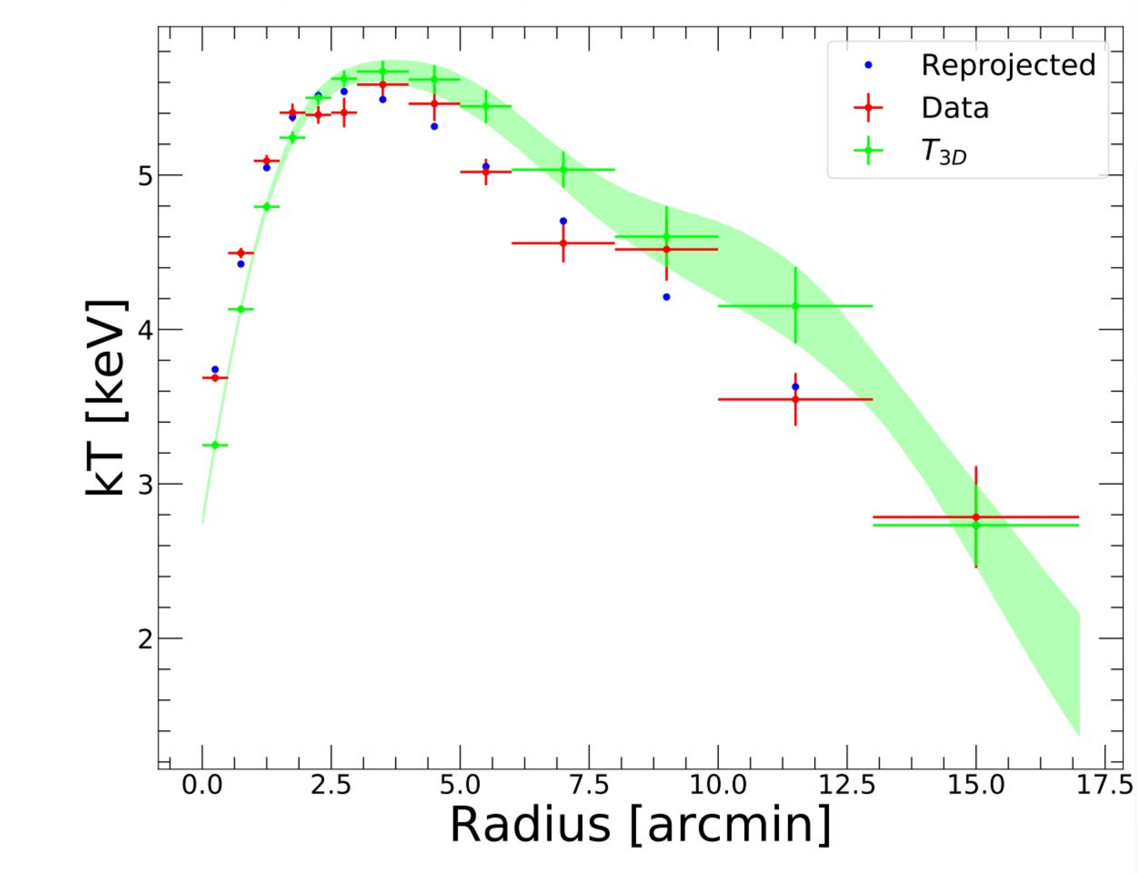
Green: 3D model obtained with PyMC3

Blue: Best-fit 3D model reprojected to compare with data

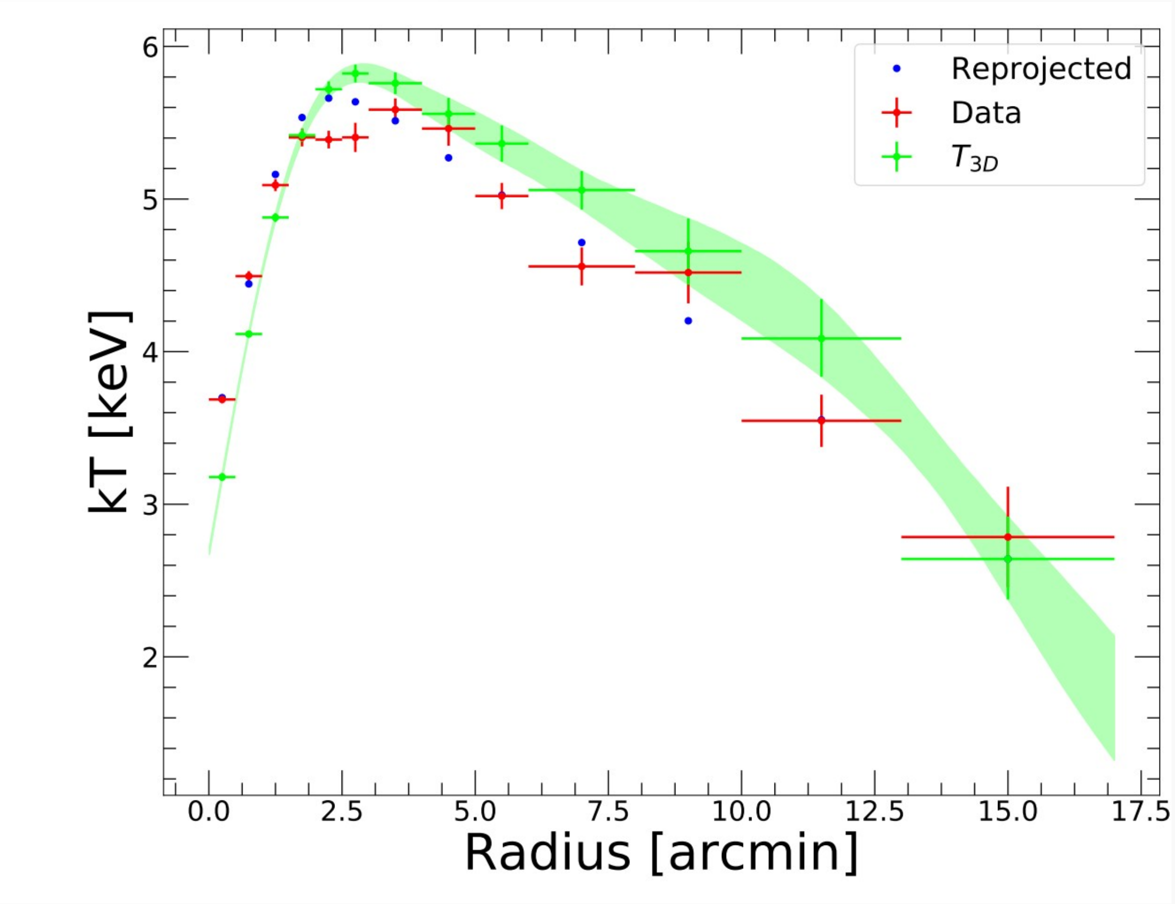
PSF convolution of temperature profiles

- In this case the relation between T_{2D} and T_{3D} is the same as before, but with a matrix $T = PSF \cdot V$

w/o PSF

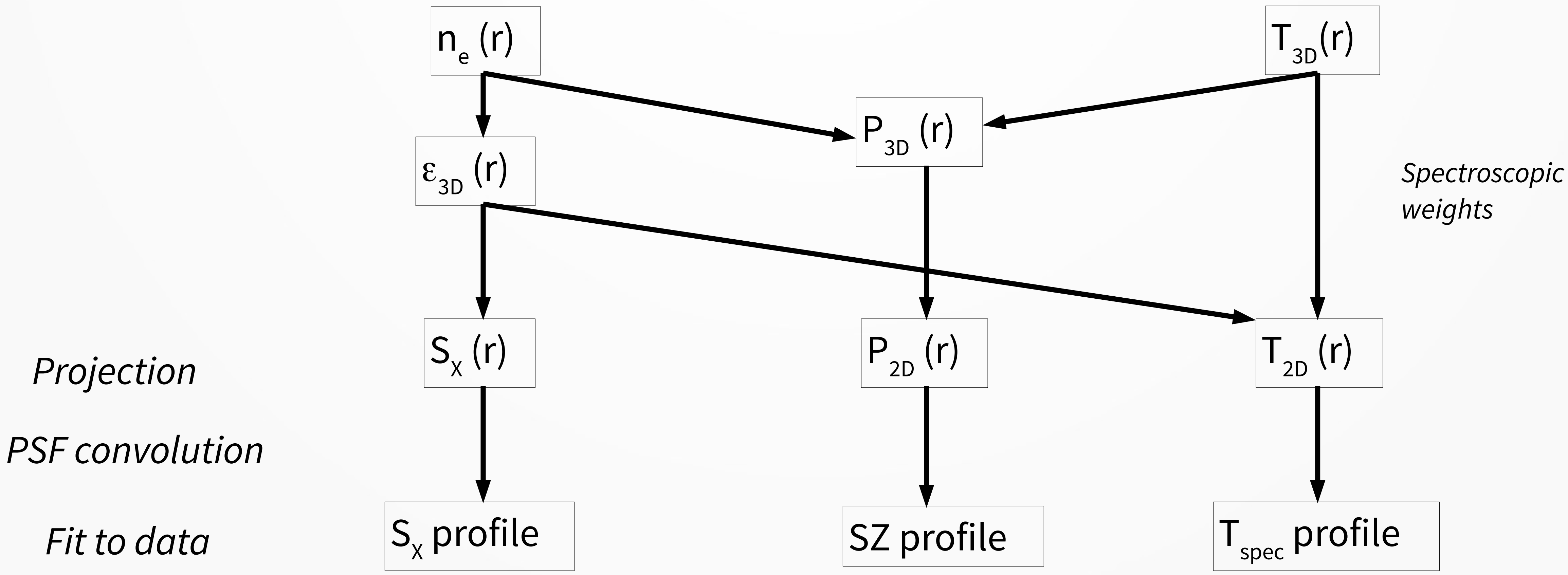


w PSF



Hydrostatic mass reconstruction

- The multiscale density and temperature models can be optimized jointly



Fitting a mass model

- If we choose to use a parametric mass model:

$$M(<r) = f(r, \theta)$$

- The "total" HSE pressure becomes

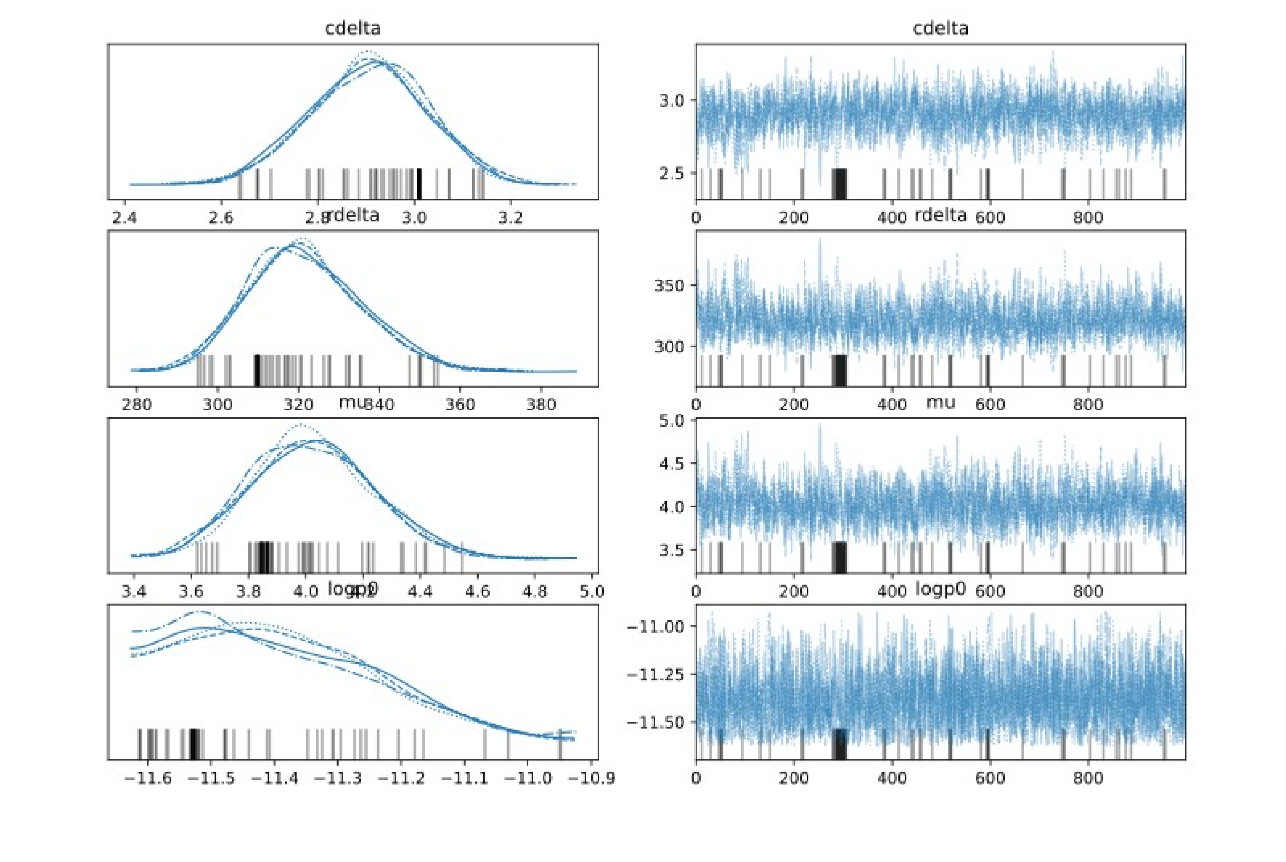
$$P(r, \theta) = P_0 + \int_r^{r_0} \frac{\rho_{gas} G f(r', \theta)}{r'^2} dr'$$

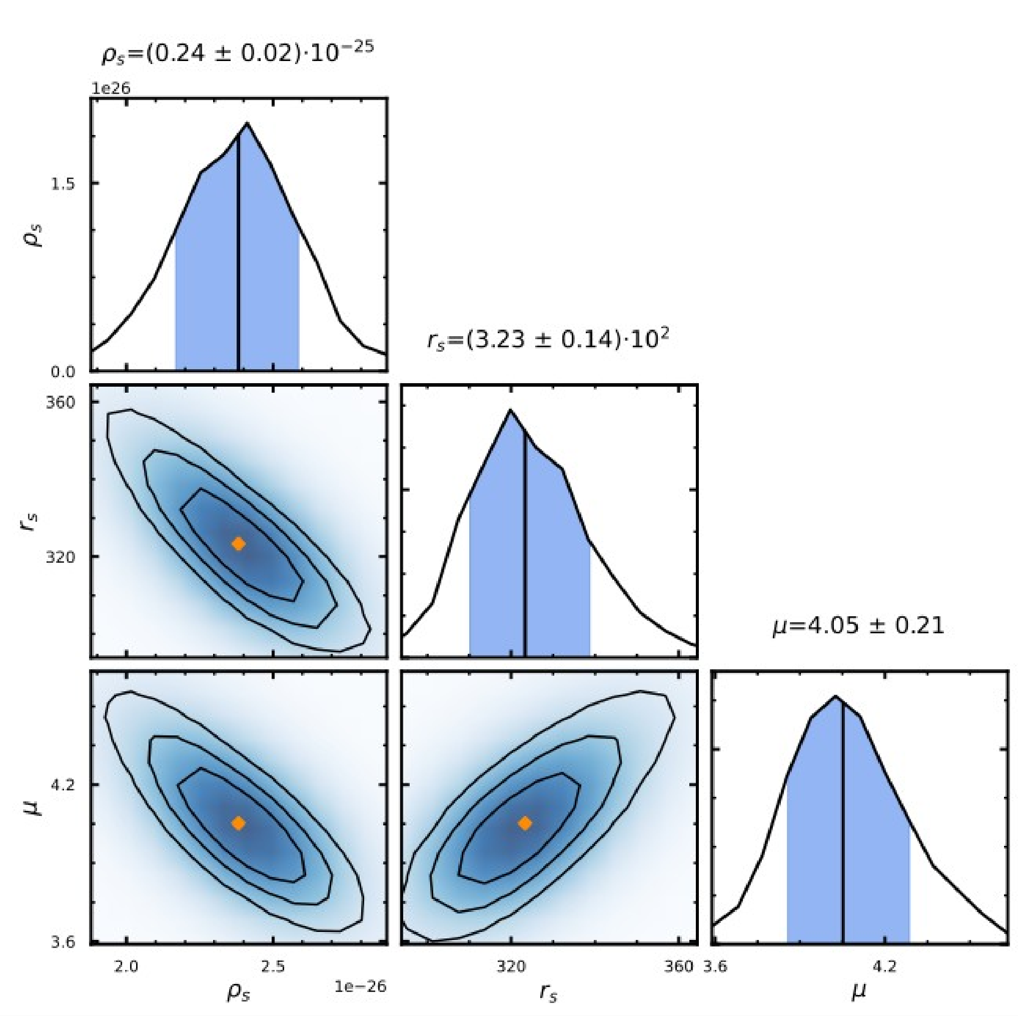
  with r_0 the outermost radius of the profile and P_0 the pressure at r_0

- At any point in the fitting process, multiscale parameters predict ρ_{gas} such that with the mass model we can predict $P(r, \theta)$
- Priors on the model parameters θ can be easily set in the code

Posterior parameter distributions

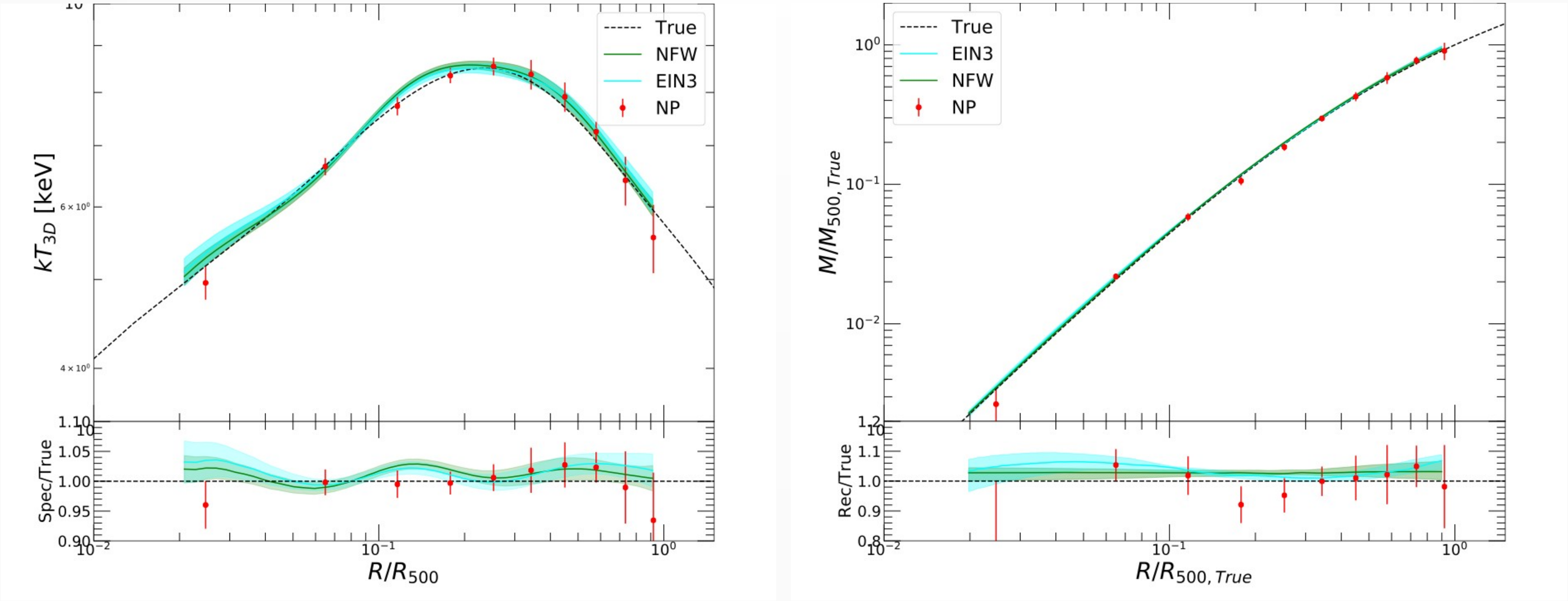
- Example posterior distributions for the Einasto fit





Tests on mock data

- We created mock XMM observations of a fiducial NFW cluster, including projection, PSF convolution, energy redistribution etc.

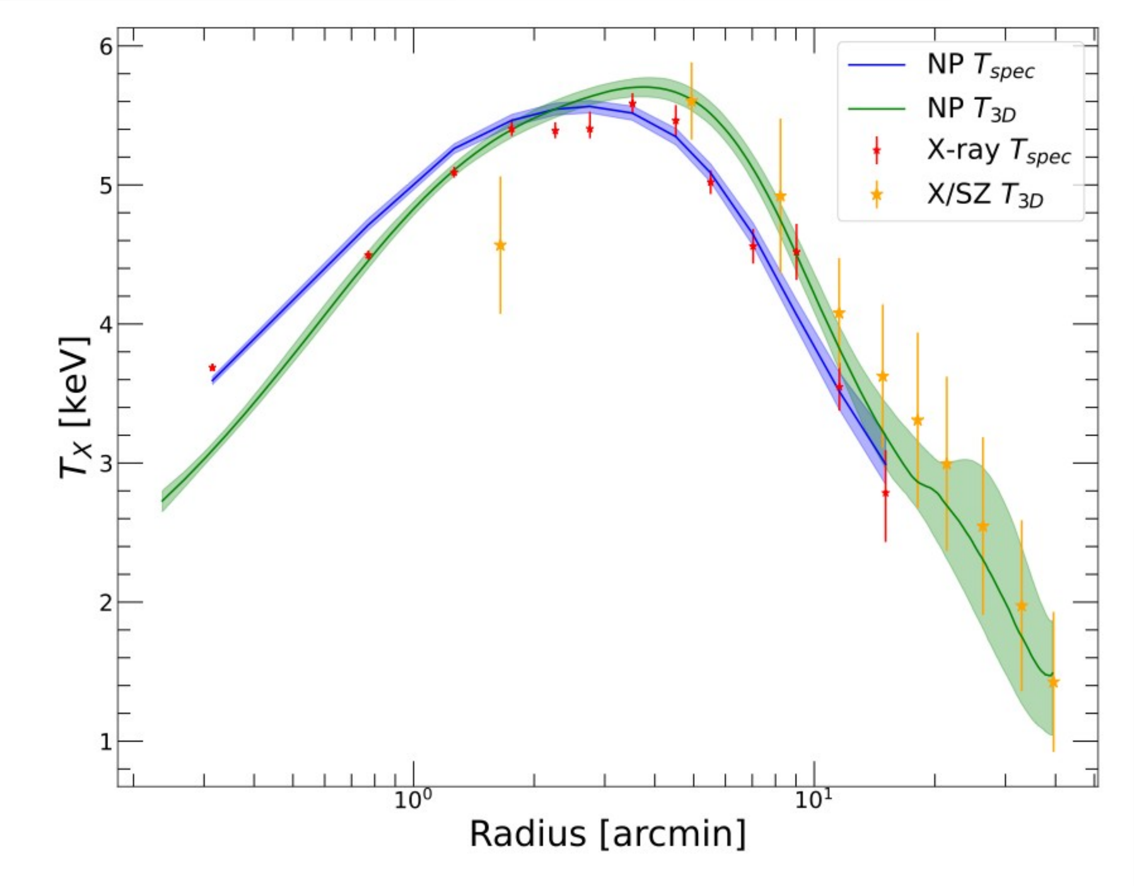


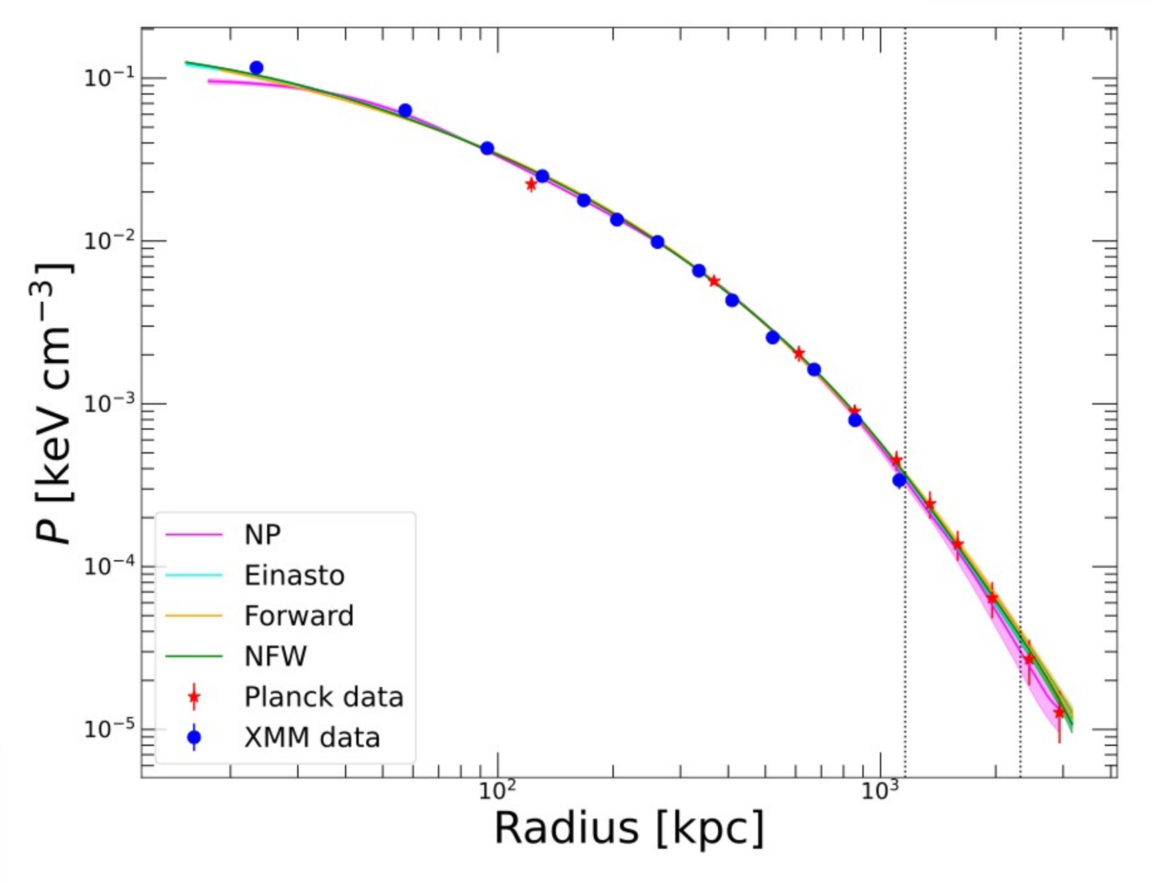
Eckert et al. 2022a

- Our code recovers the true profile with <3% accuracy

Example: application to A1795

- Joint XMM + Planck reconstruction of the 3D cluster properties

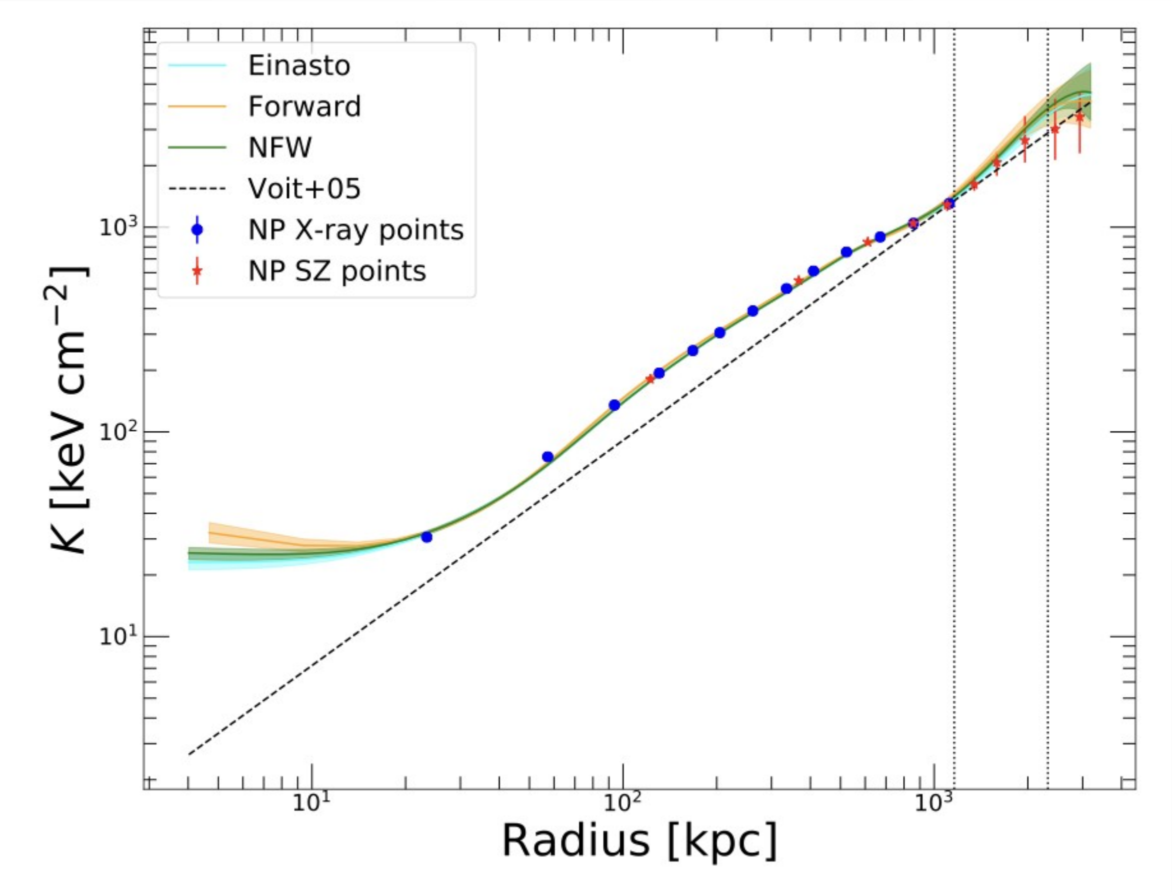


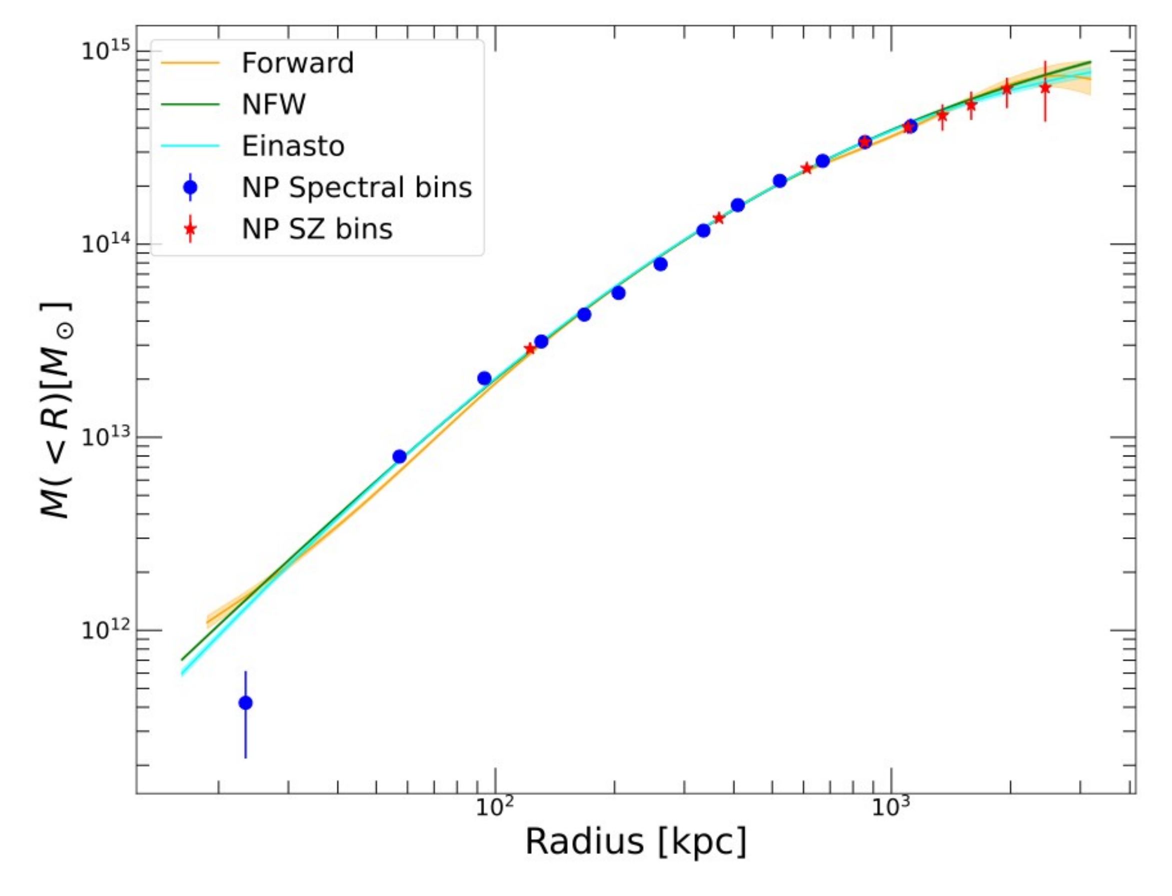


Eckert et al. 2022a

Application to A1795

- Derived thermodynamic and mass profiles





Eckert et al. 2022a

Summary

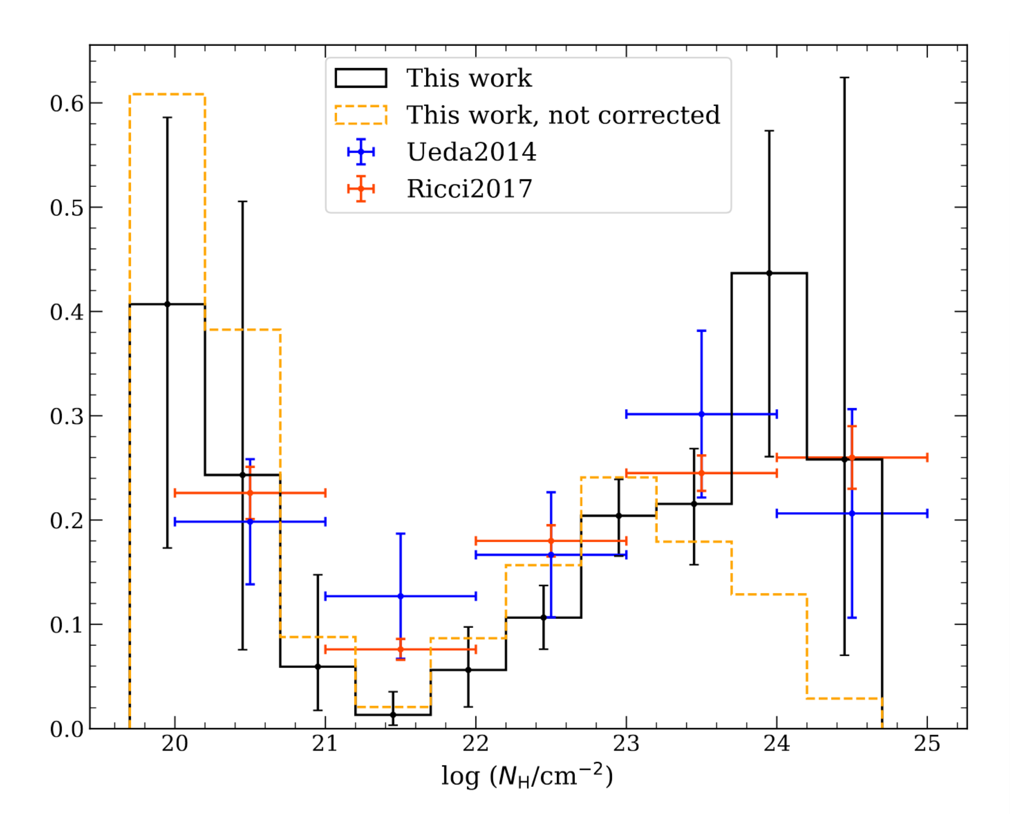
- We introduce *pyproffit* and *hydromass* , two public Python packages to reconstruct galaxy cluster properties
- The tools use multiscale decomposition to deproject and deconvolve observed profiles: King functions (density) and Gaussian processes (temperature)
- 1D PSF convolution with a mixing matrix is accurate at the sub-percent level
- S_X , T_{spec} and y_{SZ} profiles can be fitted jointly to reconstruct n_{3D}, T_{3D}
- The packages include fast Bayesian optimization using Hamiltonian Monte Carlo
- Within a single common framework *hydromass* also includes fitting with many popular mass models (e.g. NFW, Einasto), parametric forward model, and polytropic reconstruction
- Tests using mock data show that the method is accurate at the <3% level
- Extensive documentation is already in place for *pyproffit* and will be there soon for *hydromass*. Please try them out and give us feedback!

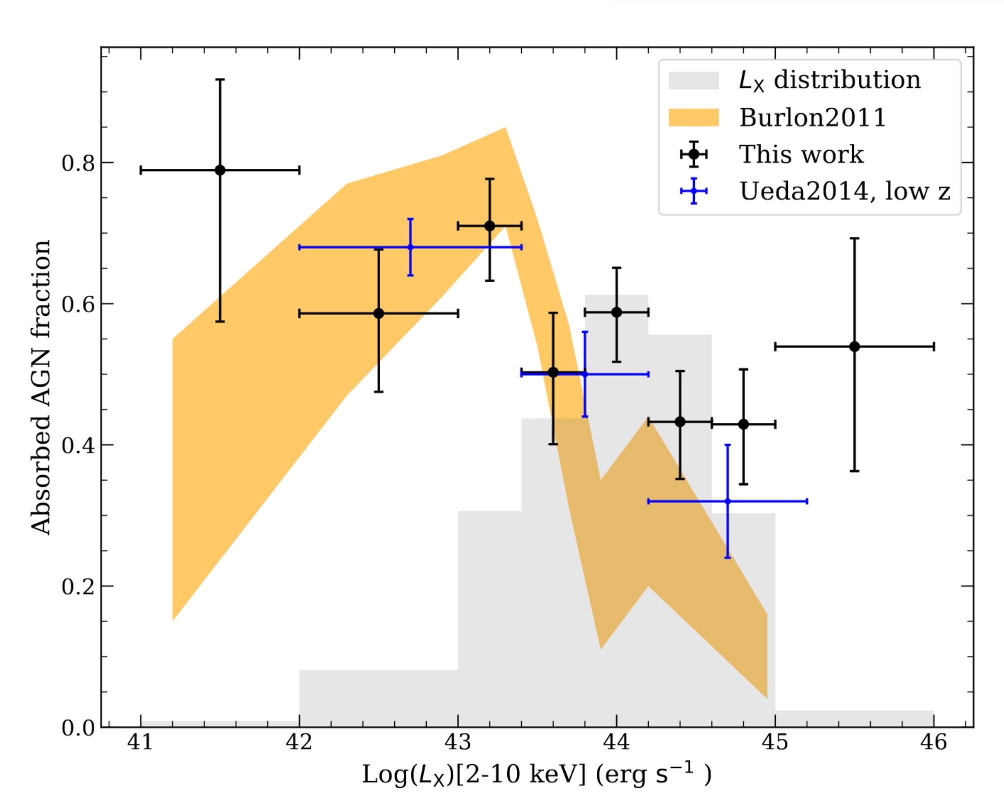
Work in progress

- Joint fit with weak lensing data
- Add non-thermal pressure modeling
- Constrain line-of-sight elongation and 3D structure
- Marginalize over the position of the center
- Test reconstruction with mock observations of hydrodynamical simulations

Bonus: AGN spectral parameters

- We recently presented a Bayesian approach to the reconstruction of AGN spectral parameters from X-ray survey data





Ge et al. 2022 a,b

ArXiv: 2111.14925 and 2111.15235

Choice of parameters

- Given a temperature profile with N points measured at radii $\{r_i\}$, we set P=100 Gaussians with means logarithmically spaced between the center and the

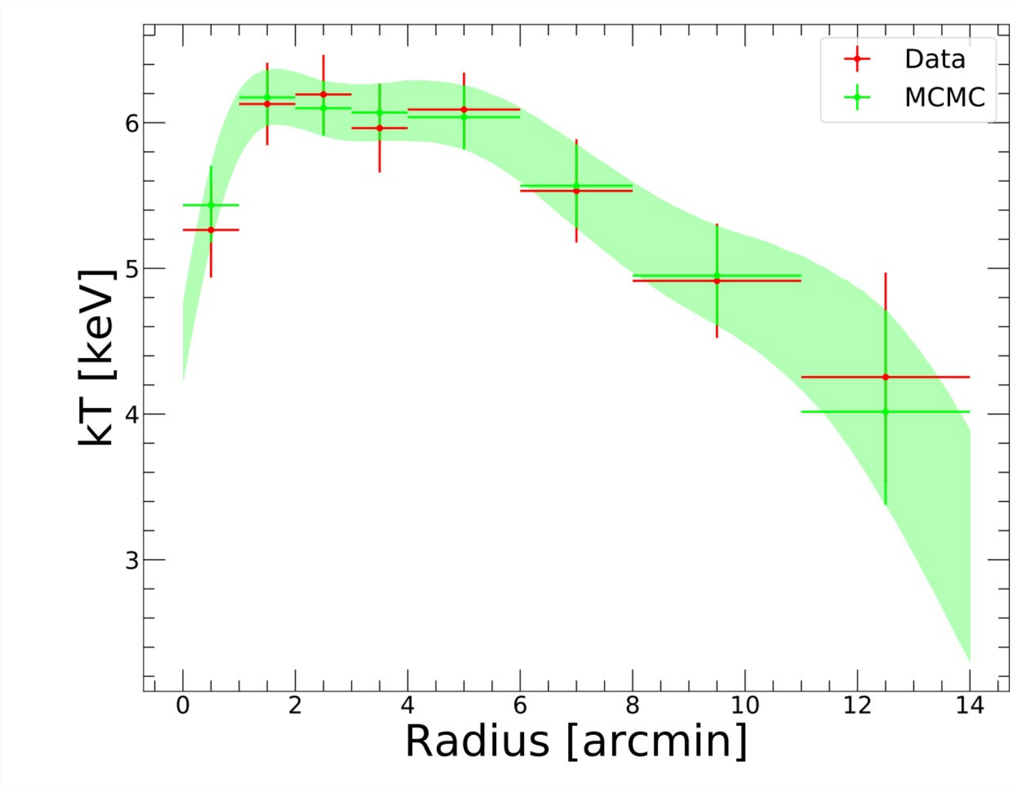
$$\log \mu_j = \log r_0 + j\left(\log r_{max} - \log r_0\right)$$

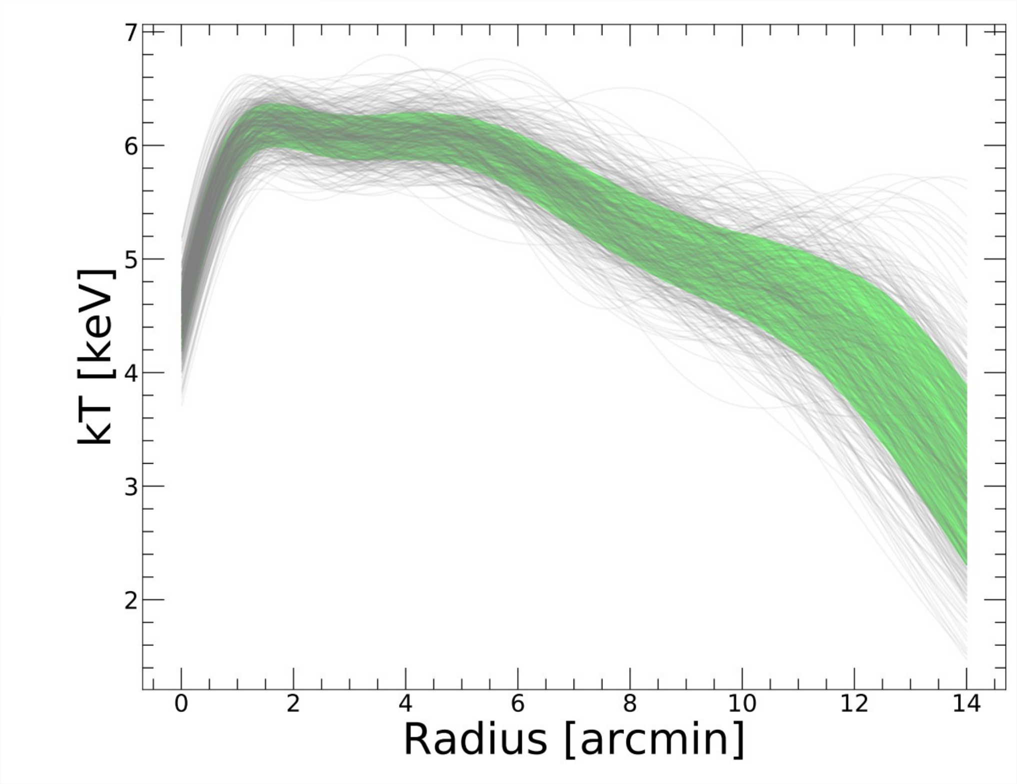
- And standard deviations set to the bin size in order to kill fluctuations on a scale smaller than the binning

- The values of $\{\sigma_j\}$ can be tuned to achieve more/less smoothing

Implementation on 2D profiles

- First I started by testing how well the model can reproduce the shape of 2D profiles
- Optimization performed using PyMC3 (Hamiltonian Monte Carlo)





Now let's move to 3D

- We set $V_{i,j}$ the volume of (spherical) shell j projected onto (cylindrical) shell i, then the 2D temperature is given by

$$T_{2D}(r_i) = \frac{\sum_{j=1}^{N} V_{i,j} w_j T_{3D}(r_j)}{\sum_{j=1}^{N} V_{i,j} w_j}$$

  with w_j the spectroscopic-like weights. Here we use Mazzotta et al. (2004) weights,

$$w_j = EM_{3D}(r_j) T_{3D}(r_j)^{-3/4}$$

- It is easy to substitute here another function for the weights and to include the PSF matrix, since only the relation between T_{2D} and T_{3D} is changed; otherwise the problem is the same