

DM_halo_models

Generated by Doxygen 1.9.4

1 DM_halo_models	1
1.1 Overview	1
1.2 Requirements	1
1.3 Fitting	1
1.4 Acknowledgements	2
1.5 Citations	2
1.6 Contact	2
2 Namespace Index	3
2.1 Namespace List	3
3 Class Index	5
3.1 Class List	5
4 Namespace Documentation	7
4.1 alp_funcs Namespace Reference	7
4.1.1 Detailed Description	8
4.2 alp_params Namespace Reference	8
4.2.1 Detailed Description	8
4.2.2 Function Documentation	9
4.2.2.1 alphamatched()	9
4.3 cdm_funcs Namespace Reference	9
4.3.1 Detailed Description	10
4.4 cdm_params Namespace Reference	10
4.4.1 Detailed Description	11
4.5 constants Namespace Reference	11
4.5.1 Detailed Description	11
4.5.2 Function Documentation	12
4.5.2.1 standard()	12
4.6 galaxy Namespace Reference	12
4.6.1 Detailed Description	12
4.7 halo Namespace Reference	13
4.7.1 Detailed Description	13
4.8 model_fit Namespace Reference	13
4.8.1 Detailed Description	14
4.8.2 Function Documentation	14
4.8.2.1 abund_match_rel()	15
4.8.2.2 BTFR()	16
4.8.2.3 conc_mass_rel_Du()	16
4.8.2.4 conc_mass_rel_Wa()	17
4.9 results Namespace Reference	17
4.9.1 Detailed Description	18
5 Class Documentation	19

5.1 alp_funcs.base_funcs Class Reference	19
5.1.1 Detailed Description	20
5.1.2 Constructor & Destructor Documentation	20
5.1.2.1 __init__()	20
5.1.3 Member Function Documentation	20
5.1.3.1 mass_sol_init()	21
5.1.3.2 msol()	21
5.1.3.3 rc()	22
5.1.3.4 rhoc()	22
5.1.3.5 xc()	23
5.1.4 Member Data Documentation	23
5.1.4.1 cdmhalo	23
5.1.4.2 matched	24
5.1.4.3 mfree	24
5.2 cdm_funcs.base_funcs Class Reference	24
5.2.1 Detailed Description	25
5.2.2 Constructor & Destructor Documentation	25
5.2.2.1 __init__()	25
5.2.3 Member Function Documentation	25
5.2.3.1 mass_frac_dc14()	25
5.2.3.2 rc()	26
5.2.3.3 rhoc()	27
5.2.3.4 xc()	27
5.2.4 Member Data Documentation	28
5.2.4.1 mass_num	28
5.3 cdm_params.base_funcs Class Reference	28
5.3.1 Detailed Description	28
5.3.2 Constructor & Destructor Documentation	28
5.3.2.1 __init__()	28
5.3.3 Member Function Documentation	29
5.3.3.1 v200min_dc14()	29
5.3.3.2 v200min_frac_dc14check()	30
5.4 constants.fitting_dict Class Reference	30
5.4.1 Detailed Description	31
5.4.2 Constructor & Destructor Documentation	31
5.4.2.1 __init__()	32
5.4.3 Member Function Documentation	33
5.4.3.1 args_opts()	33
5.4.3.2 params_vals()	34
5.5 galaxy.galaxy Class Reference	36
5.5.1 Detailed Description	36
5.5.2 Constructor & Destructor Documentation	36

5.5.2.1 <code>__init__()</code>	36
5.5.3 Member Data Documentation	37
5.5.3.1 <code>data</code>	37
5.5.3.2 <code>df</code>	37
5.5.3.3 <code>df1</code>	38
5.6 <code>model_fit.grar_fit</code> Class Reference	38
5.6.1 Detailed Description	39
5.6.2 Constructor & Destructor Documentation	39
5.6.2.1 <code>__init__()</code>	39
5.6.3 Member Function Documentation	39
5.6.3.1 <code>fit()</code>	40
5.6.3.2 <code>g_bar()</code>	40
5.6.3.3 <code>g_rar()</code>	41
5.6.3.4 <code>g_tot()</code>	41
5.6.3.5 <code>grar_model()</code>	42
5.7 <code>cdm_funcs.halo</code> Class Reference	43
5.7.1 Detailed Description	43
5.7.2 Constructor & Destructor Documentation	43
5.7.2.1 <code>__init__()</code>	44
5.7.3 Member Function Documentation	44
5.7.3.1 <code>mass()</code>	44
5.7.3.2 <code>Mvir()</code>	45
5.7.4 Member Data Documentation	46
5.7.4.1 <code>mass_num</code>	46
5.8 <code>cdm_params.halo</code> Class Reference	46
5.8.1 Detailed Description	47
5.8.2 Constructor & Destructor Documentation	47
5.8.2.1 <code>__init__()</code>	47
5.8.3 Member Function Documentation	48
5.8.3.1 <code>params()</code>	48
5.8.4 Member Data Documentation	48
5.8.4.1 <code>data</code>	48
5.8.4.2 <code>params_vals</code>	49
5.9 <code>halo.halo</code> Class Reference	49
5.9.1 Detailed Description	50
5.9.2 Constructor & Destructor Documentation	50
5.9.2.1 <code>__init__()</code>	50
5.9.3 Member Function Documentation	50
5.9.3.1 <code>mass()</code>	50
5.9.3.2 <code>velocity()</code>	51
5.9.4 Member Data Documentation	52
5.9.4.1 <code>data</code>	52

5.9.4.2 halo_init	52
5.9.4.3 params	52
5.9.4.4 params_vals	52
5.10 model_fit.model_fit Class Reference	53
5.10.1 Detailed Description	53
5.10.2 Constructor & Destructor Documentation	53
5.10.2.1 __init__()	53
5.10.3 Member Function Documentation	54
5.10.3.1 bic()	54
5.10.3.2 fit()	54
5.10.3.3 plot()	55
5.10.3.4 residual()	56
5.10.3.5 velocity_tot()	56
5.10.4 Member Data Documentation	57
5.10.4.1 args_opts	57
5.11 results.plots Class Reference	57
5.11.1 Detailed Description	59
5.11.2 Constructor & Destructor Documentation	59
5.11.2.1 __init__()	60
5.11.3 Member Function Documentation	60
5.11.3.1 BIC_CDM()	60
5.11.3.2 BIC_diffs_CDM()	61
5.11.3.3 BIC_psi_mfix_ex()	62
5.11.3.4 BIC_psi_mfree()	63
5.11.3.5 chi_box_Einasto_checks()	64
5.11.3.6 chi_CDM()	64
5.11.3.7 chi_dist_CDM()	65
5.11.3.8 chi_dist_CDM_checks()	66
5.11.3.9 chi_dist_DC14_checks()	66
5.11.3.10 chi_dist_Einasto_checks()	67
5.11.3.11 chi_dist_psi_mfix_ex()	68
5.11.3.12 chi_dist_psi_mfree()	68
5.11.3.13 chi_gal_psi_mfix()	69
5.11.3.14 chi_psi_mfix()	70
5.11.3.15 chi_psi_mfix_ex()	71
5.11.3.16 chi_psi_mfree()	72
5.11.3.17 MLd_degen_CDM()	73
5.11.3.18 Msol_psi_mfix()	74
5.11.3.19 Msol_psi_mfix_ex()	75
5.11.3.20 Msol_psi_mfree()	75
5.11.3.21 params_dist_CDM()	77
5.11.3.22 params_dist_CDM_checks()	78

5.11.3.23 params_dist_DC14_checks()	78
5.11.3.24 params_dist_Einasto_checks()	79
5.11.3.25 params_dist_psi_mfix_ex()	80
5.11.3.26 params_dist_psi_mfree()	80
5.11.3.27 params_scatter_Einasto_checks()	81
5.11.3.28 relations_CDM()	82
5.11.3.29 relations_psi_mfix_ex()	82
5.11.3.30 relations_psi_mfree()	83
5.11.3.31 rotcurves_CDM_all()	84
5.11.3.32 rotcurves_CDM_check()	85
5.11.3.33 rotcurves_DC14_check()	85
5.11.3.34 rotcurves_Einasto_check()	86
5.11.3.35 rotcurves_psi_all()	86
5.12 results.results_CDM_all Class Reference	87
5.12.1 Detailed Description	88
5.12.2 Constructor & Destructor Documentation	88
5.12.2.1 __init__()	88
5.12.3 Member Function Documentation	89
5.12.3.1 fit()	89
5.13 results.results_CDM_check Class Reference	90
5.13.1 Detailed Description	90
5.13.2 Constructor & Destructor Documentation	90
5.13.2.1 __init__()	90
5.13.3 Member Function Documentation	91
5.13.3.1 fit()	91
5.14 results.results_DC14_check Class Reference	92
5.14.1 Detailed Description	92
5.14.2 Constructor & Destructor Documentation	92
5.14.2.1 __init__()	92
5.14.3 Member Function Documentation	93
5.14.3.1 fit()	93
5.15 results.results_Einasto_check Class Reference	93
5.15.1 Detailed Description	94
5.15.2 Constructor & Destructor Documentation	94
5.15.2.1 __init__()	94
5.15.3 Member Function Documentation	95
5.15.3.1 fit()	95
5.16 results.results_psi_multi_all Class Reference	96
5.16.1 Detailed Description	96
5.16.2 Constructor & Destructor Documentation	96
5.16.2.1 __init__()	96
5.16.3 Member Function Documentation	97

5.16.3.1 fit()	97
5.16.4 Member Data Documentation	98
5.16.4.1 cdmhalo	98
5.17 results.results_psi_single_all Class Reference	98
5.17.1 Detailed Description	99
5.17.2 Constructor & Destructor Documentation	99
5.17.2.1 __init__()	99
5.17.3 Member Function Documentation	100
5.17.3.1 fit()	100
5.17.4 Member Data Documentation	101
5.17.4.1 cdmhalo	101
5.18 alp_funcs.soliton Class Reference	101
5.18.1 Detailed Description	102
5.18.2 Constructor & Destructor Documentation	102
5.18.2.1 __init__()	102
5.18.3 Member Function Documentation	103
5.18.3.1 mass()	103
5.18.3.2 Mhalo()	103
5.18.3.3 Mvir()	104
5.18.4 Member Data Documentation	104
5.18.4.1 cdmhalo	104
5.18.4.2 matched	105
5.18.4.3 mfree	105
5.19 alp_params.soliton Class Reference	105
5.19.1 Detailed Description	106
5.19.2 Constructor & Destructor Documentation	106
5.19.2.1 __init__()	107
5.19.3 Member Function Documentation	107
5.19.3.1 params()	107
5.19.4 Member Data Documentation	108
5.19.4.1 cdmhalo	108
5.19.4.2 data	108
5.19.4.3 matched	108
5.19.4.4 mfree	108
5.19.4.5 params_vals	108

Chapter 1

DM_halo_models

Testing DM models with SPARC

1.1 Overview

This repository was created to test several galactic dark matter (DM) models against the Spitzer Photometry and Accurate Rotation Curves (SPARC) catalog data (see <http://astroweb.case.edu/SPARC/>). It can be used to obtain fit results for each of the DM models analyzed, as well as calculate various properties such as the galactic DM mass, etc. For more information pertaining to the math and physics behind the calculations, see <https://arxiv.org/abs/2204.01871>.

1.2 Requirements

See requirements.txt

1.3 Fitting

All results, including figures, can be reproduced using the Jupyter notebooks. The main results of the paper are included in the notebooks :

- fits_CDM_all.ipynb
- fits_Einasto.ipynb

Comparisons to previous studies can be found in the notebooks :

- checks_CDM_all.ipynb
- checks_DC14_NFW.ipynb
- checks_Einasto_NFW.ipynb

1.4 Acknowledgements

This material is based upon work supported by the U.S. Department of Energy (DOE), Office of Science, Office of Workforce Development for Teachers and Scientists, Office of Science Graduate Student Research (SCGSR) program. The SCGSR program is administered by the Oak Ridge Institute for Science and Education (ORISE) for the DOE. ORISE is managed by ORAU under contract number DE-SC0014664. All opinions expressed in this paper are the authors' and do not necessarily reflect the policies and views of DOE, ORAU, or ORISE.

Thanks to Joshua Eby and Peter Suranyi for valuable discussions and comments. Thanks to Mike Sokoloff and Daniel Vieira for setting up computational resources to be used in the next installment of this study.

1.5 Citations

Below is the bibtex formatted citation for <https://arxiv.org/abs/2204.01871>,

```
@article{Street:2022nib, author = "Street, Lauren and Gnedin, Nickolay Y. and Wijewardhana, L. C. R.", title = "{↵ Testing multiflavored ULDM models with SPARC}", eprint = "2204.01871", archivePrefix = "arXiv", primaryClass = "astro-ph.CO", reportNumber = "FERMILAB-PUB-22-268-T", month = "4", year = "2022" }
```

Below is the bibtex formatted citation for this repository,

```
@misc{Street_DM_halo_models, author = {Street, Lauren and Gnedin, Nickolay Y. and Wijewardhana, L. C. R.}, title = {{DM halo models}}, url = { https://github.com/laurenstreet/DM\_halo\_models } }
```

1.6 Contact

streetlg@mail.uc.edu

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

alp_funcs	The package containing the functions for the ULDM halo models	7
alp_params	The package containing the parameters for the ULDM halo models	8
cdm_funcs	The package containing the functions for the CDM halo models	9
cdm_params	The package containing the parameters for the CDM halo models	10
constants	The package containing constants, fitting routine definitions, and fitting routine parameter values	11
galaxy	The package containing information about all SPARC catalog galaxies	12
halo	The package containing the general information for all DM halo models	13
model_fit	The package containing the general fitting procedures	13
results	The package containing the procedures to obtain results for various cases	17

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

alp_funcs.base_funcs	The class containing base functions needed to describe the ULDM halo models	19
cdm_funcs.base_funcs	The class containing base functions needed to describe the CDM halo models	24
cdm_params.base_funcs	The class containing all base functions for the CDM halo parameters	28
constants.fitting_dict	The class containing dictionaries of fitting variables	30
galaxy.galaxy	The class containing data for all SPARC catalog galaxies	36
model_fit.grar_fit	The class containing the fitting procedure for the gravitational acceleration relation	38
cdm_funcs.halo	The class containing mass functions for the CDM halo models	43
cdm_params.halo	The class containing all parameters for fitting the CDM halo models	46
halo.halo	The class containing definitions to describe a DM halo	49
model_fit.model_fit	The class containing the fitting procedure assuming a chosen halo model for a chosen galaxy .	53
results.plots	The class to obtain rotation curve plots for given galaxies	57
results.results_CDM_all	The class to obtain fit results for all CDM halos analyzed	87
results.results_CDM_check	The class to obtain fit results to check all CDM model implementation	90
results.results_DC14_check	The class to obtain fit results to check the DC14 and NFW model implementation	92
results.results_Einasto_check	The class to obtain fit results to check the Einasto and NFW model implementation	93
results.results_psi_multi_all	The class to obtain fit results for all double flavored ULDM models analyzed	96
results.results_psi_single_all	The class to obtain fit results for all single flavored ULDM models analyzed	98
alp_funcs.soliton	The class containing mass functions for the ULDM halo models	101
alp_params.soliton	The class containing all parameters for fitting the ULDM halo models	105

Chapter 4

Namespace Documentation

4.1 alp_funcs Namespace Reference

The package containing the functions for the ULDM halo models.

Classes

- class [base_funcs](#)
The class containing base functions needed to describe the ULDM halo models.
- class [soliton](#)
The class containing mass functions for the ULDM halo models.

Variables

- **standard_consts_in** = [constants.standard\(\)](#)
dictionary
Dictionary containing standard constants.
- def **kmTOGeV** = [standard_consts_in](#)['kmTOGeV']
float
Conversion from km to GeV
- def **sTOGeV** = [standard_consts_in](#)['sTOGeV']
float
Conversion from s to GeV
- def **kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']
float
Conversion from kpc to GeV
- def **msun** = [standard_consts_in](#)['msun']
float
Mass of sun in GeV
- def **MP** = [standard_consts_in](#)['MP']
float
Planck mass in GeV
- def **rhocrit** = [standard_consts_in](#)['rhocrit']
float
Critical density of universe in GeV^4
- **fitting_dict_in** = [constants.fitting_dict\(\)](#)
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class using all default values

4.1.1 Detailed Description

The package containing the functions for the ULDM halo models.

4.2 alp_params Namespace Reference

The package containing the parameters for the ULDM halo models.

Classes

- class [soliton](#)

The class containing all parameters for fitting the ULDM halo models.

Functions

- def [alphamatched](#) (m22, msol, c200, V200)

Define the value of α when using ULDM matched models.

Variables

- **standard_consts_in** = [constants.standard](#)()

dictionary

Dictionary containing standard constants.

- def **kmTOGeV** = [standard_consts_in](#)['kmTOGeV']

float

Conversion from km to GeV

- def **sTOGeV** = [standard_consts_in](#)['sTOGeV']

float

Conversion from s to GeV

- def **kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']

float

Conversion from kpc to GeV

- def **msun** = [standard_consts_in](#)['msun']

float

Mass of sun in GeV

- def **MP** = [standard_consts_in](#)['MP']

float

Planck mass in GeV

- def **rhoc** = [standard_consts_in](#)['rhocrit']

float

Critical density of universe in GeV^4

- **fitting_dict_in** = [constants.fitting_dict](#)()

[constants.fitting_dict](#) instance

Instance of the [constants.fitting_dict](#) class using all default values

4.2.1 Detailed Description

The package containing the parameters for the ULDM halo models.

4.2.2 Function Documentation

4.2.2.1 alphamatched()

```
def alp_params.alphamatched (
    m22,
    msol,
    c200,
    V200 )
```

Define the value of α when using ULDM matched models.

This defines the value of α in the ULDM matched models. This value of is fixed from the condition $\rho_{\text{sol}}(3r_{\text{c,sol}}) = \rho_{\text{Einasto}}(3r_{\text{c,sol}})$.

Parameters

<i>m22</i>	mass of ULDM in units of 10^{-22} eV
<i>msol</i>	mass of soliton in units of solar masses
<i>c200</i>	Einasto profile variable c_{200}
<i>V200</i>	Einasto profile variable V_{200}

Returns

float Value of α to be used for ULDM matched models

4.3 cdm_funcs Namespace Reference

The package containing the functions for the CDM halo models.

Classes

- class [base_funcs](#)
The class containing base functions needed to describe the CDM halo models.
- class [halo](#)
The class containing mass functions for the CDM halo models.

Variables

- **standard_consts_in** = [constants.standard\(\)](#)
dictionary
Dictionary containing standard constants.
- **def kmTOGeV** = [standard_consts_in](#)['kmTOGeV']
float
Conversion from km to GeV
- **def sTOGeV** = [standard_consts_in](#)['sTOGeV']
float
Conversion from s to GeV
- **def kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']
float
Conversion from kpc to GeV
- **def msun** = [standard_consts_in](#)['msun']
float
Mass of sun in GeV
- **def MP** = [standard_consts_in](#)['MP']
float
Planck mass in GeV
- **def rhocrit** = [standard_consts_in](#)['rhocrit']
float
Critical density of universe in GeV^4
- **fitting_dict_in** = [constants.fitting_dict\(\)](#)
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class using all default values

4.3.1 Detailed Description

The package containing the functions for the CDM halo models.

4.4 cdm_params Namespace Reference

The package containing the parameters for the CDM halo models.

Classes

- class [base_funcs](#)
The class containing all base functions for the CDM halo parameters.
- class [halo](#)
The class containing all parameters for fitting the CDM halo models.

Variables

- **standard_consts_in** = [constants.standard\(\)](#)
dictionary
Dictionary containing standard constants.
- def **kmTOGeV** = [standard_consts_in](#)['kmTOGeV']
float
Conversion from km to GeV
- def **sTOGeV** = [standard_consts_in](#)['sTOGeV']
float
Conversion from s to GeV
- def **kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']
float
Conversion from kpc to GeV
- def **msun** = [standard_consts_in](#)['msun']
float
Mass of sun in GeV
- def **MP** = [standard_consts_in](#)['MP']
float
Planck mass in GeV
- def **rhoc** = [standard_consts_in](#)['rhocrit']
float
Critical density of universe in GeV^4
- **fitting_dict_in** = [constants.fitting_dict\(\)](#)
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class using all default values

4.4.1 Detailed Description

The package containing the parameters for the CDM halo models.

4.5 constants Namespace Reference

The package containing constants, fitting routine definitions, and fitting routine parameter values.

Classes

- class [fitting_dict](#)
The class containing dictionaries of fitting variables.

Functions

- def [standard](#) ()
The dictionary containing standard constants.

4.5.1 Detailed Description

The package containing constants, fitting routine definitions, and fitting routine parameter values.

4.5.2 Function Documentation

4.5.2.1 standard()

```
def constants.standard ( )
```

The dictionary containing standard constants.

This defines the dictionary containing standard constants needed during fitting procedure.

Returns

dictionary

Dictionary of standard constants:

- kmTOGeV in units $[\text{km}^{-1} \text{GeV}^{-1}]$ = conversion from kilometer to GeV
- sTOGeV in units $[\text{s}^{-1} \text{GeV}^{-1}]$ = conversion from second to GeV
- kpcTOGeV in units $[\text{kpc}^{-1} \text{GeV}^{-1}]$ = conversion from kpc to GeV
- msun in units $[\text{GeV}]$ = mass of sun in GeV
- MP in units $[\text{GeV}]$ = Planck mass in GeV
- rhocrit in units $[\text{GeV}^4]$ = critical density of universe in GeV^4
- MOND_gdag in units $[\text{m s}^{-2}]$ = gravitational acceleration constant for gravitational acceleration relation in MOND theory

4.6 galaxy Namespace Reference

The package containing information about all SPARC catalog galaxies.

Classes

- class [galaxy](#)

The class containing data for all SPARC catalog galaxies.

4.6.1 Detailed Description

The package containing information about all SPARC catalog galaxies.

All data has been taken from SPARC database : <http://astroweb.case.edu/SPARC/>
Data files can be found in:

- DM_halos_models/pyfiles/data/MassModels_Lelli2016c.txt
- DM_halos_models/pyfiles/data/SPARC_Lelli2016c.txt

4.7 halo Namespace Reference

The package containing the general information for all DM halo models.

Classes

- class [halo](#)
The class containing definitions to describe a DM halo.

Variables

- **standard_consts_in** = [constants.standard\(\)](#)
dictionary
Dictionary containing standard constants.
- def **kmTOGeV** = [standard_consts_in](#)['kmTOGeV']
float
Conversion from km to GeV
- def **sTOGeV** = [standard_consts_in](#)['sTOGeV']
float
Conversion from s to GeV
- def **kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']
float
Conversion from kpc to GeV
- def **msun** = [standard_consts_in](#)['msun']
float
Mass of sun in GeV
- def **MP** = [standard_consts_in](#)['MP']
float
Planck mass in GeV
- **fitting_dict_in** = [constants.fitting_dict\(\)](#)
constants.fitting_dict instance
Instance of the [constants.fitting_dict](#) class using all default values

4.7.1 Detailed Description

The package containing the general information for all DM halo models.

4.8 model_fit Namespace Reference

The package containing the general fitting procedures.

Classes

- class [grar_fit](#)
The class containing the fitting procedure for the gravitational acceleration relation.
- class [model_fit](#)
The class containing the fitting procedure assuming a chosen halo model for a chosen galaxy.

Functions

- def [conc_mass_rel_Du](#) (m200)
Define the concentration-mass relation (CMR).
- def [conc_mass_rel_Wa](#) (m200)
Define the concentration-mass relation (CMR).
- def [BTFR](#) (Vf)
Define the baryonic Tully-Fisher relation (BTFR).
- def [abund_match_rel](#) (m200)
Define the abundance matching relation (AMR).

Variables

- **standard_consts_in** = [constants.standard](#)()
dictionary
Dictionary containing standard constants.
- def **kmTOGeV** = [standard_consts_in](#)['kmTOGeV']
float
Conversion from km to GeV
- def **sTOGeV** = [standard_consts_in](#)['sTOGeV']
float
Conversion from s to GeV
- def **kpcTOGeV** = [standard_consts_in](#)['kpcTOGeV']
float
Conversion from kpc to GeV
- def **msun** = [standard_consts_in](#)['msun']
float
Mass of sun in GeV
- def **MP** = [standard_consts_in](#)['MP']
float
Planck mass in GeV
- **fitting_dict_in** = [constants.fitting_dict](#)()
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class using all default values

4.8.1 Detailed Description

The package containing the general fitting procedures.

4.8.2 Function Documentation

4.8.2.1 abund_match_rel()

```
def model_fit.abund_match_rel (
    m200 )
```

Define the abundance matching relation (AMR).

This defines the AMR for a given halo mass. See Peter S. Behroozi et al 2013 ApJ 770 57 : <https://doi.org/10.1088/0004-637x/770/1/57> and Peter S. Behroozi et al 2013 ApJL 762 L31 : <https://doi.org/10.1088/2041-8205/762/2/L31> for more information. The AMR is given by,

$$\frac{M_*}{M_{200}} = 2N \left[\left(\frac{M_{200}}{M_1 M_\odot} \right)^{-\beta} + \left(\frac{M_{200}}{M_1 M_\odot} \right)^{-\gamma} \right]^{-1},$$

where $N = 0.0351$, $\beta = 1.375$, $\gamma = 0.608$, and $\log_{10}(M_1) = 11.59$.

Parameters

<i>m200</i>	float Total mass of the DM halo in units of solar masses.
-------------	--

Returns

float
Total stellar mass of a galaxy in units of solar masses.

4.8.2.2 BTFR()

```
def model_fit.BTFR (
    Vf )
```

Define the baryonic Tully-Fisher relation (BTFR).

This defines the BTFR for a given maximum circular velocity of a galaxy. See Federico Lelli et al 2016 ApJL 816 L14 : <https://doi.org/10.3847/2041-8205/816/1/L14> for more information. The BTFR is given by,

$$\log_{10} \frac{M_b}{M_{\odot}} = s \log_{10} \left(\frac{V_f}{\text{km s}^{-1}} \right) + \log_{10} A,$$

where M_{\odot} is the mass of sun, $s = 3.71 \pm 0.08$, and $\log_{10} A = 2.27 \pm 0.18$.

Parameters

<i>Vf</i>	float Maximum circular velocity in units of km s^{-1} .
-----------	---

Returns

float
Total baryonic mass of a galaxy in units of solar masses.

4.8.2.3 conc_mass_rel_Du()

```
def model_fit.conc_mass_rel_Du (
    m200 )
```

Define the concentration-mass relation (CMR).

This defines the CMR for a given halo mass. See Monthly Notices of the Royal Astronomical Society, Volume 441, Issue 4, 11 July 2014, Pages 3359–3374 : <https://doi.org/10.1093/mnras/stu742> for more information. The CMR is given by,

$$\log_{10} c_{200} = 0.905 - 0.101 \log_{10} \left(\frac{M_{200}}{10^{12} h^{-1} M_{\odot}} \right),$$

where $h^{-1} = 0.73$ and M_{\odot} is the mass of sun.

Parameters

<i>m200</i>	float Total mass of the DM halo in units of solar masses.
-------------	--

Returns

float
Concentration parameter assuming the CMR.

4.8.2.4 conc_mass_rel_Wa()

```
def model_fit.conc_mass_rel_Wa (
    m200 )
```

Define the concentration-mass relation (CMR).

This defines the CMR for a given halo mass. See Nature volume 585, pages 39–42 (2020) : <https://doi.org/10.1038/s41586-020-2642-9> for more information. The CMR is given by,

$$c_{200} = \sum_{i=0}^5 c_i \ln \left(\frac{M_{200}}{h^{-1} M_{\odot}} \right)^i,$$

where $h^{-1} = 0.6777$, M_{\odot} is the mass of sun, $c_0 = 27.112$, $c_1 = -0.381$, $c_2 = -1.853 \times 10^{-3}$, $c_3 = -4.141 \times 10^{-4}$, and $c_5 = 3.208 \times 10^{-7}$.

Parameters

<i>m200</i>	float Total mass of the DM halo in units of solar masses.
-------------	--

Returns

float
Concentration parameter assuming the CMR.

4.9 results Namespace Reference

The package containing the procedures to obtain results for various cases.

Classes

- class [plots](#)
The class to obtain rotation curve plots for given galaxies.
- class [results_CDM_all](#)
The class to obtain fit results for all CDM halos analyzed.

- class [results_CDM_check](#)
The class to obtain fit results to check all CDM model implementation.
- class [results_DC14_check](#)
The class to obtain fit results to check the DC14 and NFW model implementation.
- class [results_Einasto_check](#)
The class to obtain fit results to check the Einasto and NFW model implementation.
- class [results_psi_multi_all](#)
The class to obtain fit results for all double flavored ULDM models analyzed.
- class [results_psi_single_all](#)
The class to obtain fit results for all single flavored ULDM models analyzed.

Variables

- `fitting_dict_in = constants.fitting_dict()`
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class using all default values

4.9.1 Detailed Description

The package containing the procedures to obtain results for various cases.

Chapter 5

Class Documentation

5.1 alp_funcs.base_funcs Class Reference

The class containing base functions needed to describe the ULDM halo models.

Public Member Functions

- `def __init__ (self, model, fit_dict_in=fitting_dict_in)`
Define the constructor of the [base_funcs](#) class.
- `def msol (self, params)`
Define the total soliton mass.
- `def rc (self, params)`
Define the soliton profile variable rc.
- `def rhoc (self, params)`
Define the soliton profile variable rhoc.
- `def xc (self, params, r)`
Define the soliton profile variable xc.
- `def mass_sol_init (self, params, r)`
Define the mass profile of the soliton.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of [constants.fitting_dict.args_opts](#)
- **matched**
bool
Denotes how to combine the soliton and outer halo.
- **mfree**

- float*
 Denotes how to treat soliton particle mass in fitting procedure.
- [cdmhalo](#)
 - float*
 Denotes which CDM profile to use for outer halo in ULDM galactic structure.
- **m22**
 - float*
 Soliton particle mass in units of 10^{-22} eV.
- **m22_2**
 - float*
 Soliton particle mass two in units of 10^{-22} eV.

5.1.1 Detailed Description

The class containing base functions needed to describe the ULDM halo models.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 `__init__()`

```
def alp_funcs.base_funcs.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the [base_funcs](#) class.

This defines the constructor of the [base_funcs](#) class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • <code>psi_single</code> • <code>psi_multi</code>
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.1.3 Member Function Documentation

5.1.3.1 mass_sol_init()

```
def alp_funcs.base_funcs.mass_sol_init (
    self,
    params,
    r )
```

Define the mass profile of the soliton.

This defines the mass of the soliton at a given radius for the given model parameters.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N] (for single flavored models) or ndarray[2,N] (for double flavored models)
Numpy array of N soliton mass values (for single flavored models) or numpy array of [2,N] mass values (for double flavored models) in units of solar mass.

5.1.3.2 msol()

```
def alp_funcs.base_funcs.msol (
    self,
    params )
```

Define the total soliton mass.

This defines the total soliton mass for the given model parameters.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .

Returns

float (for single flavored models) or ndarray[2] (for double flavored models)
The total soliton mass (for single flavored models) or both total soliton masses, msol_1 and msol_2 (for double flavored models) in units of solar mass.

5.1.3.3 rc()

```
def alp_funcs.base_funcs.rc (
    self,
    params )
```

Define the soliton profile variable rc.

This defines the soliton profile variable rc for the given model parameters. The soliton profile variable rc is given by,

$$r_c \approx 0.228 \left(\frac{M_{\text{sol}}}{10^9} \right)^{-1} m^{-2},$$

where $M_{\text{sol}} = \text{alp_funcs.base_funcs.msol}$ and $m = \text{m22}$ in [alp_params.soliton.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .

Returns

float (for single flavored models) or ndarray[2] (for double flavored models)

The halo profile variable rc in units of kpc. The soliton profile variable rc (for single flavored models) or both soliton profile variables, rc_1 and rc_2 (for double flavored models) in units of kpc.

5.1.3.4 rhoc()

```
def alp_funcs.base_funcs.rhoc (
    self,
    params )
```

Define the soliton profile variable rhoc.

This defines the soliton profile variable rhoc for the given model parameters. The soliton profile variables rhoc is given by,

$$\rho_c \approx 7 \times 10^9 \left(\frac{M_{\text{sol}}}{10^9} \right)^4 m^6,$$

where $M_{\text{sol}} = \text{alp_funcs.base_funcs.msol}$ and $m = \text{m22}$ in [alp_params.soliton.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .

Returns

float (for single flavored models) or ndarray[2] (for double flavored models)

The soliton profile variable rhoc (for single flavored models) or both soliton profile variables, rhoc_1 and rhoc_2 (for double flavored models) in units of M_{\odot}/kpc^3 .

5.1.3.5 xc()

```
def alp_funcs.base_funcs.xc (
    self,
    params,
    r )
```

Define the soliton profile variable xc.

This defines the soliton profile variable $xc = r/r_c$ for the given model parameters. The soliton profile variable xc is given by,

$$x_c = r/r_c,$$

where $r_c = \text{alp_funcs.base_funcs.rc}$.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N] (for single flavored models) or ndarray[2,N] (for double flavored models)

Numpy array of N (for N given radii) soliton profile variables xc (for single flavored models) or [2,N] soliton profile variables, xc_1 and xc_2 (for double flavored models).

5.1.4 Member Data Documentation**5.1.4.1 cdmhalo**

```
alp_funcs.base_funcs.cdmhalo
```

float

Denotes which CDM profile to use for outer halo in ULDM galactic structure.

Equal to constants.fitting_dict.sol_cdmhalo

5.1.4.2 matched

`alp_funcs.base_funcs.matched`

bool

Denotes how to combine the soliton and outer halo.

Equal [constants.fitting_dict.sol_match](#)

5.1.4.3 mfree

`alp_funcs.base_funcs.mfree`

float

Denotes how to treat soliton particle mass in fitting procedure.

Equal to [constants.fitting_dict.sol_mfree](#)

The documentation for this class was generated from the following file:

- `pyfiles/models/alps/alp_funcs.py`

5.2 cdm_funcs.base_funcs Class Reference

The class containing base functions needed to describe the CDM halo models.

Public Member Functions

- `def __init__ (self, model, fit_dict_in=fitting_dict_in)`
Define the constructor of the [base_funcs](#) class.
- `def mass_frac_dc14 (self, params)`
Define the mass fraction for the DC14 model.
- `def rc (self, params)`
Define the halo profile variable rc.
- `def rhoc (self, params)`
Define the halo profile variable rhoc.
- `def xc (self, params, r)`
Define the halo profile variable xc.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of [constants.fitting_dict.args_opts](#)
- **mass_num**
int
Used to differentiate between soliton 1 and soliton 2 in double flavored ULDM models.

5.2.1 Detailed Description

The class containing base functions needed to describe the CDM halo models.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 `__init__()`

```
def cdm_funcs.base_funcs.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the `base_funcs` class.

This defines the constructor of the `base_funcs` class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert • DC14 • Einasto • NFW
<i>fit_dict_in</i>	(optional) <code>constants.fitting_dict</code> instance Instance of the <code>constants.fitting_dict</code> class. Contains all the necessary rules and values to be used in fitting.

5.2.3 Member Function Documentation

5.2.3.1 `mass_frac_dc14()`

```
def cdm_funcs.base_funcs.mass_frac_dc14 (
    self,
    params )
```

Define the mass fraction for the DC14 model.

This defines the stellar to DM mass fraction for the DC14 model assuming the given model parameters. The stellar to DM mass fraction is given by,

$$X = \log_{10} (M_*/M_{\text{halo}}) ,$$

where,

$$M_* \approx \left(\tilde{\Upsilon}_{\text{disk}} + \tilde{\Upsilon}_{\text{bulge}} \right) L,$$

with L = Luminosity in [galaxy.galaxy.data](#), $M_{\text{halo}} = \text{cdm_funcs.halo.Mvir}$, and $\tilde{\Upsilon}_{\text{disk}} = \text{MLD}$ and $\tilde{\Upsilon}_{\text{bulge}} = \text{MLB}$ in [cdm_params.halo.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .

Returns

float

The stellar to DM mass fraction in the DC14 model.

5.2.3.2 rc()

```
def cdm_funcs.base_funcs.rc (
    self,
    params )
```

Define the halo profile variable rc.

This defines the halo profile variable rc for the given model parameters. The halo profile variable rc is given by,

$$r_c = \sqrt{\frac{3}{2\pi \rho_{\text{crit}}} \frac{M_P V_{200}}{20 c_{200}}},$$

where $\rho_{\text{crit}} = \text{rhocrit}$ and $M_P = \text{MP}$ in [constants.standard](#), while $V_{200} = \text{v200}$ and $c_{200} = \text{c200}$ in [cdm_params.halo.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .

Returns

float

The halo profile variable rc in units of kpc.

5.2.3.3 rhoc()

```
def cdm_funcs.base_funcs.rhoc (
    self,
    params )
```

Define the halo profile variable rhoc.

This defines the halo profile variable rhoc for the given model parameters. The halo profile variables rhoc is given by,

$$\rho_c = \frac{M_{200}}{4\pi r_c^3 \left[\ln(1 + c_{200}) - \frac{c_{200}}{1+c_{200}} \right]},$$

where $r_c = \text{cdm_funcs.base_funcs.rc}$, $M_{200} = \text{cdm_funcs.halo.Mvir}$, and $c_{200} = \text{cdm_params.halo.params.c200}$ in [cdm_params.halo.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .

Returns

float

The halo profile variable rhoc in units of M_\odot/kpc^3 .

5.2.3.4 xc()

```
def cdm_funcs.base_funcs.xc (
    self,
    params,
    r )
```

Define the halo profile variable xc.

This defines the halo profile variable xc for the given model parameters. The halo profile variable xc is given by,

$$x_c = r/r_c,$$

where $r_c = \text{cdm_funcs.base_funcs.rc}$.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N]
Numpy array of N halo profile variables xc.

5.2.4 Member Data Documentation**5.2.4.1 mass_num**

`cdm_funcs.base_funcs.mass_num`

int

Used to differentiate between soliton 1 and soliton 2 in double flavored ULDM models.

Outer CDM halo is halo 1 if `mass_num = 0` and is halo 2 if `mass_num = 1`.

The documentation for this class was generated from the following file:

- `pyfiles/models/cdm/cdm_funcs.py`

5.3 cdm_params.base_funcs Class Reference

The class containing all base functions for the CDM halo parameters.

Public Member Functions

- `def __init__(self)`
Define the constructor of the `base_funcs` class.
- `def v200min_frac_dc14check(self, mstar, mgas, vfac)`
Define the expression for V200 to be assumed in particular cases.
- `def v200min_dc14(self, mstar)`
Define the minimum allowed V200 for the DC14 model.

5.3.1 Detailed Description

The class containing all base functions for the CDM halo parameters.

This class contains functions necessary to describe CDM halo parameters for various cases.

5.3.2 Constructor & Destructor Documentation**5.3.2.1 __init__()**

```
def cdm_params.base_funcs.__init__(
    self )
```

Define the constructor of the `base_funcs` class.

This defines the constructor of the `base_funcs` class.

Parameters

<i>self</i>	object pointer
-------------	----------------

5.3.3 Member Function Documentation

5.3.3.1 v200min_dc14()

```
def cdm_params.base_funcs.v200min_dc14 (
    self,
    mstar )
```

Define the minimum allowed V200 for the DC14 model.

This defines the minimum allowed V200 for the DC14 model. This is assumed when using the fit routines :

- `constants.fitting_dict.fit_routine = 'uni_priors'`,
- `constants.fitting_dict.fit_routine = 'c200_priors_check'`,
- `constants.fitting_dict.fit_routine = 'v200_priors_check'`,
- `constants.fitting_dict.fit_routine = 'MLd_priors_check'`,
- `constants.fitting_dict.fit_routine = 'MLb_priors_check'`,
- `constants.fitting_dict.fit_routine = 'CDM_check'` The allowed values for V200 are given by,

$$V_{200} \geq \left[10^{1.3} M_* \sqrt{\frac{2\pi \rho_{\text{crit}}}{3}} \frac{20 \times 10^9}{M_P^3} \right]^{1/3},$$

where,

$$M_* \approx \left(\tilde{\Upsilon}_{\text{disk}} + \tilde{\Upsilon}_{\text{bulge}} \right) L.$$

Here, $\rho_{\text{crit}} = \text{rhocrit}$ and $M_P = \text{MP}$ in `constants.standard`, $M_{\text{gas}} = \text{Mgas}$ and $L = \text{Luminosity}$ in `galaxy.galaxy.data`, $\tilde{\Upsilon}_{\text{disk}} = \text{MLD}$ and $\tilde{\Upsilon}_{\text{bulge}} = \text{MLB}$ in `cdm_params.halo.params`.

Parameters

<i>self</i>	object pointer
<i>mstar</i>	float Total stellar mass in units of 10^9 solar masses.

Returns

float
Minimum allowed value for V200 for the DC14 model for various cases.

5.3.3.2 v200min_frac_dc14check()

```
def cdm_params.base_funcs.v200min_frac_dc14check (
    self,
    mstar,
    mgas,
    vfac )
```

Define the expression for V200 to be assumed in particular cases.

This defines the allowable values for V200 in \log_{10} space when using the fit routine `constants.fitting_dict.fit_routine = 'DC14_check'`. The allowable values for V200 are given by,

$$V_{200} \geq \left[(M_* + M_{\text{gas}}) \sqrt{\frac{2\pi \rho_{\text{crit}}}{3}} \frac{20 \times 10^9}{0.2 M_P^3} \right]^{1/3},$$

where,

$$M_* \approx (\tilde{\Upsilon}_{\text{disk}} + \tilde{\Upsilon}_{\text{bulge}}) L.$$

Here, $\rho_{\text{crit}} = \text{rhocrit}$ and $M_P = \text{MP}$ in `constants.standard`, $M_{\text{gas}} = \text{Mgas}$ and $L = \text{Luminosity}$ in `galaxy.galaxy.data`, $\tilde{\Upsilon}_{\text{disk}} = \text{MLD}$ and $\tilde{\Upsilon}_{\text{bulge}} = \text{MLB}$ in `cdm_params.halo.params`.

Parameters

<i>self</i>	object pointer
<i>mstar</i>	float Total stellar mass in units of 10^9 solar masses.
<i>mgas</i>	float Total HI gas mass in units of 10^9 solar masses.
<i>vfac</i>	float Factor to vary in fitting procedure.

Returns

float
Value for V200 in \log_{10} space in units of $\log_{10} (\text{km s}^{-1})$.

The documentation for this class was generated from the following file:

- `pyfiles/models/cdm/cdm_params.py`

5.4 constants.fitting_dict Class Reference

The class containing dictionaries of fitting variables.

Public Member Functions

- `def __init__ (self, fit_routine='uni_priors', sol_match=False, sol_mfree=False, sol_cdm_halo='Einasto', sol_m22=10**(1.5), sol_m22_2=1, sol_m22_tab=np.float_power(10,(np.arange(0, 2, 0.15))), sol_m22_tab_prime=np.float_power(10,(np.arange(-3, 4, 0.5))), sol_m22_2_tab=np.float_power(10,(np.arange(0, 2, 0.15))), sol_m22_2_tab_prime=np.float_power(10,(np.arange(-3, 4, 0.5))), sol_mass_ind=0)`

- Define the constructor of the `fitting_dict` class.
- def `args_opts` (self)
Define the fitting rules.
- def `params_vals` (self)
Define the fitting values.

Public Attributes

- **fit_routine**
str
Denotes the fitting routine to use.
- **sol_match**
bool
Denotes how to combine the soliton and outer halo.
- **sol_mfree**
bool
Denotes how to treat soliton particle mass in fitting procedure.
- **sol_cdm_halo**
str
Denotes which CDM profile to use for outer halo in ULDM galactic structure.
- **sol_m22**
float
Soliton particle mass (for single flavored models) and soliton 1 particle mass (for double flavored models) in units of 10^{-22} eV.
- **sol_m22_2**
float
Soliton 2 particle mass (for double flavored models) in units of 10^{-22} eV.
- **sol_m22_tab**
ndarray[N]
Numpy array of soliton particle masses (for single flavored, matched models) and soliton 1 particle masses (for double flavored, matched models) in units of 10^{-22} eV.
- **sol_m22_tab_prime**
ndarray[N]
Numpy array of soliton particle masses (for single flavored, summed models) and soliton 1 particle masses (for double flavored, summed models) in units of 10^{-22} eV.
- **sol_m22_2_tab**
ndarray[N]
Numpy array of soliton 2 particle masses (for double flavored, matched models) in units of 10^{-22} eV.
- **sol_m22_2_tab_prime**
ndarray[N]
Numpy array of soliton 2 particle masses (for double flavored, summed models) in units of 10^{-22} eV.
- **sol_mass_ind**
int
Used to differentiate between soliton 1 and soliton 2 in double flavored models.

5.4.1 Detailed Description

The class containing dictionaries of fitting variables.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 `__init__()`

```
def constants.fitting_dict.__init__ (
    self,
    fit_routine = 'uni_priors',
    sol_match = False,
    sol_mfree = False,
    sol_cdm_halo = 'Einasto',
    sol_m22 = 10**(1.5),
    sol_m22_2 = 1,
    sol_m22_tab = np.float_power(10, (np.arange(0, 2, 0.15))),
    sol_m22_tab_prime = np.float_power(10, (np.arange(-3, 4, 0.5))),
    sol_m22_2_tab = np.float_power(10, (np.arange(0, 2, 0.15))),
    sol_m22_2_tab_prime = np.float_power(10, (np.arange(-3, 4, 0.5))),
    sol_mass_ind = 0 )
```

Define the constructor of the `fitting_dict` class.

This defines the constructor of the `fitting_dict` class.

Parameters

<i>self</i>	object pointer
<i>fit_routine</i>	<p>(optional) str Denotes the fitting routine to use. Can be equal to:</p> <ul style="list-style-type: none"> • 'uni_priors' : uniform priors on all parameters (used for main results of paper) • 'c200_priors_check' : used to check affect of changing prior ranges for c200 • 'v200_priors_check' : used to check affect of changing prior ranges for v200 • 'MLd_priors_check' : used to check affect of changing prior ranges for MLd • 'MLb_priors_check' : used to check affect of changing prior ranges for MLb • 'CDM_check' : used to obtain results to be compared with Pengfei Li et al 2020 ApJS 247 31 : https://doi.org/10.3847/1538-4365/ab700e • 'DC14_check' : used to obtain results to be compared with Monthly Notices of the Royal Astronomical Society, Volume 466, Issue 2, April 2017, Pages 1648–1668 : https://doi.org/10.1093/mnras/stw3101 • 'Einasto_check' : used to obtain results to be compared with Nicolas Loizeau and Glennys R. Farrar 2021 ApJL 920 L10 : https://doi.org/10.3847/2041-8213/ac1bb7
<i>sol_match</i>	<p>(optional) bool Denotes how to combine the soliton and outer halo. If equal to:</p> <ul style="list-style-type: none"> • False : soliton and CDM halo to be summed • True : soliton and CDM halo to be matched
<i>sol_mfree</i>	<p>(optional) bool Denotes how to treat soliton particle mass in fitting procedure. If equal to:</p> <ul style="list-style-type: none"> • False : soliton particle mass set to be fixed • True : soliton particle mass to be free

Parameters

<i>sol_cdm_halo</i>	(optional) str Denotes which CDM profile to use for outer halo in ULDM galactic structure. Can be equal to: <ul style="list-style-type: none"> • 'Burkert' • 'DC14' • 'Einasto' • 'NFW'
<i>sol_m22</i>	(optional) float Soliton particle mass (for single flavored models) and soliton 1 particle mass (for double flavored models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_m22_2</i>	(optional) float Soliton 2 particle mass (for double flavored models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_m22_tab</i>	(optional) ndarray[N] Numpy array of soliton particle masses (for single flavored, matched models) and soliton 1 particle masses (for double flavored, matched models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_m22_tab_prime</i>	(optional) ndarray[N] Numpy array of soliton particle masses (for single flavored, summed models) and soliton 1 particle masses (for double flavored, summed models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_m22_2_tab</i>	(optional) ndarray[N] Numpy array of soliton 2 particle masses (for double flavored, matched models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_m22_2_tab_prime</i>	(optional) ndarray[N] Numpy array of soliton 2 particle masses (for double flavored, summed models) in units of 10^{-22} eV. Only used for case in which soliton particle mass is fixed (i.e. constants.fitting_dict.sol_mfree = False)
<i>sol_mass_ind</i>	(optional) int Used to differentiate between soliton 1 and soliton 2 in double flavored models. Can be equal to 0 (to denote soliton 1) or 1 (to denote soliton 2)

5.4.3 Member Function Documentation

5.4.3.1 args_opts()

```
def constants.fitting_dict.args_opts (
    self )
```

Define the fitting rules.

This defines the rules to be assumed during fitting.

Many prior cases and fitting routines to compare to previous studies are taken into account.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of rules to be assumed during fitting:

- 'fitting_routine' : Dictionary of possible fitting routines
Each of the following fitting routines can be equal to true or false. By default, the only fitting routine that is set to true is the one equal to fit_routine.
 - 'uni_priors' : bool
 - 'c200_priors_check' : bool
 - 'v200_priors_check' : bool
 - 'MLd_priors_check' : bool
 - 'MLb_priors_check' : bool
 - 'CDM_check' : bool
 - 'DC14_check' : bool
 - 'Einasto_check' : bool
- 'soliton' : Dictionary of soliton variables
 - 'matched' : bool (equal to [constants.fitting_dict.sol_match](#))
 - 'mfree' : bool (equal to [constants.fitting_dict.sol_mfree](#))
 - 'cdm_halo' : str (equal to [constants.fitting_dict.sol_cdm_halo](#))
 - 'm22' : float (equal to [constants.fitting_dict.sol_m22](#))
 - 'm22_2' : float (equal to [constants.fitting_dict.sol_m22_2](#))
 - 'm22_tab' : ndarray[N] (equal to [constants.fitting_dict.sol_m22_tab](#))
 - 'm22_tab_prime' : ndarray[N] (equal to [constants.fitting_dict.sol_m22_tab_prime](#))
 - 'm22_2_tab' : ndarray[N] (equal to [constants.fitting_dict.sol_m22_2_tab](#))
 - 'm22_2_tab_prime' : ndarray[N] (equal to [constants.fitting_dict.sol_m22_2_tab_prime](#))
 - 'mass_ind' : int (equal to [constants.fitting_dict.sol_mass_ind](#))

5.4.3.2 params_vals()

```
def constants.fitting_dict.params_vals (
    self )
```

Define the fitting values.

This defines the values for various variables used during the fitting procedure.

Many prior cases and fitting routines to compare to previous studies are taken into account. Values are setup to be utilized in `Imfit.Parameters` class.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Possible dictionary of variables values to be used during fitting.

Variables values are setup in lists with format [starting value,min,max]

Dictionary to be used depends on value of fit_routine.

If:

- fit_routine = 'uni_priors', resulting dictionary is:
 - 'CDM' : values for variables describing CDM halos
 - * 'c200' : [3, 1, 100]
 - * 'v200' : [100, 1, 1000] in units of km s^{-1}
 - * 'v200_fac' : [1.5, 1, ∞]
 - * 'MLd' : [0.5, 0.01, 5] in units of M_{\odot}/L_{\odot}
 - * 'MLb' : [0.7, 0.01, 5] in units of M_{\odot}/L_{\odot}
 - * 'alpha' : [0.16, $-\infty$, ∞] (only used for Einasto profile)
 - 'soliton' : values for variables describing soliton
 - * 'Msol' : [10^9 , $10^{4.5}$, 10^{12}] in units of M_{\odot}
 - * 'Msol_2' : [10^9 , $10^{4.5}$, 10^{12}] in units of M_{\odot}
 - * 'm22' : [1, 10^{-3} , 10^3] in units of 10^{-22} eV
 - * 'm22_2' : [1, 10^{-3} , 10^3] in units of 10^{-22} eV
- fit_routine = 'c200_priors_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'c200' : [3, 0, ∞]
- fit_routine = 'v200_priors_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'v200' : [100, 0, ∞] in units of km s^{-1}
- fit_routine = 'MLd_priors_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'MLd' : [0.5, 0, ∞] in units of M_{\odot}/L_{\odot}
- fit_routine = 'MLb_priors_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'MLb' : [0.7, 0, ∞] in units of M_{\odot}/L_{\odot}
- fit_routine = 'CDM_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'c200' : [3, 10^{-1} , 10^3]
 - * 'v200' : [100, 10, 500] in units of km s^{-1}
- fit_routine = 'DC14_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'c200' : [$\log_{10} 5$, $\log_{10} 1$, $\log_{10} 100$]
 - * 'v200' : [$\log_{10} 100$, $\log_{10} 10$, $\log_{10} 500$] in units of $\log_{10} (\text{km s}^{-1})$
 - * 'v200_fac' : [2, 1, ∞]
 - * 'MLd' : [$\log_{10} 0.5$, $\log_{10} 0.3$, $\log_{10} 0.8$] in units of $\log_{10} (M_{\odot}/L_{\odot})$
- fit_routine = 'Einasto_check', resulting dictionary is same as 'uni_priors' with following changes:
 - 'CDM' :
 - * 'v200' : [100, 1, 500] in units of km s^{-1}
 - * 'alpha' : [0.16, 10^{-3} , 10]

The documentation for this class was generated from the following file:

- pyfiles/data_models/constants.py

5.5 galaxy.galaxy Class Reference

The class containing data for all SPARC catalog galaxies.

Public Member Functions

- `def __init__ (self, name)`
Define the constructor of the galaxy class.

Public Attributes

- **name**
str
Galaxy name
- **df**
Pandas dataframe
Dataframe that contains data from SPARC catalog.
- **df1**
Pandas dataframe
Dataframe that contains data from SPARC catalog.
- **gal_init**
Pandas dataframe
Dataframe that contains data for an initialized galaxy.
- **data**
dictionary Dictionary that contains data for the chosen galaxy (i.e.

5.5.1 Detailed Description

The class containing data for all SPARC catalog galaxies.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 __init__()

```
def galaxy.galaxy.__init__ (
    self,
    name )
```

Define the constructor of the galaxy class.

This defines the constructor of the galaxy class.

Parameters

<i>self</i>	object pointer
<i>name</i>	str
	Can be equal to : <ul style="list-style-type: none"> • Any galaxy name in the SPARC catalog (e.g. 'CamB') The galaxy with galaxy name will be analyzed.

5.5.3 Member Data Documentation

5.5.3.1 data

`galaxy.galaxy.data`

dictionary Dictionary that contains data for the chosen galaxy (i.e.

either 'name' or 'random') :

- 'Radius' : ndarray[N]
Numpy array of galactocentric radii in units of [kpc]
- 'Vobs' : ndarray[N]
Numpy array of observed circular velocities at given radii in units of $[\text{km s}^{-1}]$
- 'eVobs' : ndarray[N]
Numpy array of error in observed circular velocities in units of $[\text{km s}^{-1}]$
- 'Vgas' : ndarray[N]
Numpy array of contribution from gas to circular velocities in units of $[\text{km s}^{-1}]$
- 'Vdisk' : ndarray[N]
Numpy array of contribution from the disk to circular velocities in units of $[\text{km s}^{-1}]$
- 'Vbulge' : ndarray[N]
Numpy array of contribution from the bulge to circular velocities in units of $[\text{km s}^{-1}]$
- 'Luminosity' : float
Total luminosity at $[3.6 \mu\text{m}]$ in units of $[10^9 L_{\odot}]$
- 'Mgas' : float
Total mass of HI gas in units of $[10^9 M_{\odot}]$
- 'Vf' : float
Maximum circular velocity in units of $[\text{km s}^{-1}]$
- 'Inclination' : float
Inclination of galaxy in units of [degrees]
- 'Quality' : int
Quality flag of galaxy

5.5.3.2 df

`galaxy.galaxy.df`

Pandas dataframe

Dataframe that contains data from SPARC catalog.

Contains data from DM_halo_models/pyfiles/data/MassModels_Lelli2016c.txt

5.5.3.3 df1

`galaxy.galaxy.df1`

Pandas dataframe

Dataframe that contains data from SPARC catalog.

Contains data from DM_halo_models/pyfiles/data/SPARC_Lelli2016c.txt

The documentation for this class was generated from the following file:

- `pyfiles/data_models/galaxy.py`

5.6 model_fit.grar_fit Class Reference

The class containing the fitting procedure for the gravitational acceleration relation.

Public Member Functions

- `def __init__ (self, model, ULDM_fits=False, fit_dict_in=fitting_dict_in)`
Define the constructor of the [grar_fit](#) class.
- `def g_tot (self, params, data)`
Define the total radial gravitational acceleration of the galaxy.
- `def g_bar (self, params, data)`
Define the radial gravitational acceleration due to baryonic matter.
- `def g_rar (self, gdag, params, data)`
Define the radial gravitational acceleration relation.
- `def grar_model (self, gbarin, gdag)`
Define the general model of the radial gravitational acceleration relation.
- `def fit (self)`
Define the fit result for the modeled gravitational acceleration relation.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of [constants.fitting_dict.args_opts](#)
- **ULDM_fits**
bool
True if performing fits for the ULDM models

5.6.1 Detailed Description

The class containing the fitting procedure for the gravitational acceleration relation.

See See Stacy S. McGaugh, Federico Lelli, and James M. Schombert Phys. Rev. Lett. 117, 201101 : <https://doi.org/10.1103/PhysRevLett.117.201101> for more information

5.6.2 Constructor & Destructor Documentation

5.6.2.1 __init__()

```
def model_fit.grar_fit.__init__ (
    self,
    model,
    ULDM_fits = False,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the `grar_fit` class.

This defines the constructor of the `grar_fit` class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert • DC14 • Einasto • NFW • psi_single • psi_multi
<i>ULDM_fits</i>	(optional) bool True if performing fits for the ULDM models
<i>fit_dict_in</i>	(optional) <code>constants.fitting_dict</code> instance Instance of the <code>constants.fitting_dict</code> class. Contains all the necessary rules and values to be used in fitting.

5.6.3 Member Function Documentation

5.6.3.1 fit()

```
def model_fit.grar_fit.fit (
    self )
```

Define the fit result for the modeled gravitational acceleration relation.

This defines the fit result for the modeled gravitational acceleration relation.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

Imfit.Minimizer.minimize fit result

Fit results contain the optimized parameters and several goodness-of-fit statistics.

5.6.3.2 g_bar()

```
def model_fit.grar_fit.g_bar (
    self,
    params,
    data )
```

Define the radial gravitational acceleration due to baryonic matter.

This defines the radial gravitational acceleration due to baryonic matter assuming the given model parameters and galaxy data. The radial gravitational acceleration due to baryonic matter is given by,

$$g_{\text{bar}}(r) = V_{\text{bar}}(r)^2 / r,$$

where,

$$V_{\text{bar}}(r) = \sqrt{|V_{\text{gas}}(r)| V_{\text{gas}}(r) + \tilde{\Upsilon}_{\text{disk}} |V_{\text{disk}}(r)| V_{\text{disk}}(r) + \tilde{\Upsilon}_{\text{bulge}} |V_{\text{bulge}}(r)| V_{\text{bulge}}(r)}.$$

Here, $V_{\text{gas}}(r)$, $V_{\text{disk}}(r)$, and $V_{\text{bulge}}(r)$ can be found in [galaxy.galaxy.data](#). Finally, $\tilde{\Upsilon}_{\text{disk}} = \text{MLD}$ and $\tilde{\Upsilon}_{\text{bulge}} = \text{MLB}$ in [cdm_params.halo.params](#) (for CDM models) or [alp_params.soliton.params](#) (for ULDM models).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .

Returns

ndarray[N]

Numpy array of N values of the radial gravitational acceleration due to baryons in units of m s^{-2} .**5.6.3.3 g_rar()**

```
def model_fit.grar_fit.g_rar (
    self,
    gdag,
    params,
    data )
```

Define the radial gravitational acceleration relation.

This defines the radial gravitational acceleration relation assuming the given relation constant, gdag, model parameters and galaxy data. The radial gravitational acceleration relation is given by,

$$g_{\text{tot}}(r) = \frac{g_{\text{bar}}(r)}{1 - \exp(-\sqrt{g_{\text{bar}}(r)/g_{\dagger}})},$$

where $g_{\text{bar}}(r) = \text{model_fit.grar_fit.g_bar}$ and g_{\dagger} is some constant.

Parameters

<i>self</i>	object pointer
<i>gdag</i>	float Constant in the gravitational acceleration relation. Must be in units of m s^{-2} .
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .

Returns

ndarray[N]

Numpy array of N values of the total radial gravitational acceleration relation in units of m s^{-2} .**5.6.3.4 g_tot()**

```
def model_fit.grar_fit.g_tot (
    self,
    params,
    data )
```

Define the total radial gravitational acceleration of the galaxy.

This defines the total radial gravitational acceleration of the galaxy assuming the given model parameters and galaxy data. The total radial gravitational acceleration is given by,

$$g_{\text{tot}}(r) = V_{\text{tot}}(r)^2/r,$$

where $V_{\text{tot}}(r) = \text{model_fit.model_fit.velocity_tot}$.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .

Returns

ndarray[N]
Numpy array of N values of the total radial gravitational acceleration in units of m s^{-2} .

5.6.3.5 grar_model()

```
def model_fit.grar_fit.grar_model (
    self,
    gbarin,
    gdag )
```

Define the general model of the radial gravitational acceleration relation.

This defines the general model of the radial gravitational acceleration relation for the model parameters, gbarin and gdag. This is the form of the radial gravitational acceleration relation to be used in fitting and is given by,

$$g_{\text{tot}}(r) = \frac{g_{\text{bar}}(r)}{1 - \exp(-\sqrt{g_{\text{bar}}(r)}/g_{\dagger})}.$$

Here, the fit parameter g_{\dagger} can be found by fitting this model using $g_{\text{tot}}(r) = \text{model_fit.grar_fit.g_tot}$ and $g_{\text{bar}} = \text{model_fit.grar_fit.g_bar}$.

Parameters

<i>self</i>	object pointer
<i>gbarin</i>	ndarray[N] Numpy array of N values of baryonic gravitational accerlation in units of m s^{-2} .
<i>gdag</i>	ndarray[N] Constant in the gravitational acceleration relation in units of m s^{-2} .

Returns

ndarray[N]

Numpy array of N values of the total modeled radial gravitational acceleration in units of m s^{-2} .

The documentation for this class was generated from the following file:

- pyfiles/fitting/model_fit.py

5.7 cdm_funcs.halo Class Reference

The class containing mass functions for the CDM halo models.

Public Member Functions

- def `__init__` (self, `model`, `fit_dict_in`=`fitting_dict_in`)
Define the constructor of the halo class.
- def `mass` (self, `params`, `r`)
Define the mass profile of the CDM galactic structure.
- def `Mvir` (self, `params`)
Define the total galactic DM halo mass.

Public Attributes

- `model`
str
Model to assume for DM halo.
- `fit_dict_in`
constants.fitting_dict instance
Instance of the constants.fitting_dict class.
- `args_opts`
dictionary
Dictionary of rules to be assumed during fitting Instance of constants.fitting_dict.args_opts
- `mass_num`
int
Used to differentiate between soliton 1 and soliton 2 in double flavored ULDM models.
- `base_funcs_in`
cdm_funcs.base_funcs instance
Instance of the cdm_funcs.base_funcs class

5.7.1 Detailed Description

The class containing mass functions for the CDM halo models.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 `__init__()`

```
def cdm_funcs.halo.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the halo class.

This defines the constructor of the halo class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert • DC14 • Einasto • NFW
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.7.3 Member Function Documentation

5.7.3.1 `mass()`

```
def cdm_funcs.halo.mass (
    self,
    params,
    r )
```

Define the mass profile of the CDM galactic structure.

This defines the mass of the CDM galactic structure at a given radius given the model parameters. The mass profile depends on the model assumed. The mass profile for each model is given by :

- Burkert,

$$M_{\text{Burkert}}(r) = \pi \rho_c r_c^3 \left\{ -2 \arctan(x_c(r)) + \log \left[(1 + x_c(r))^2 (1 + x_c(r)^2) \right] \right\},$$

- DC14,

$$M_{\text{DC14}}(r) = 4\pi \rho_c r_c^3 x_c(r)^{3-\gamma} {}_2F_1 \left(\frac{3-\gamma}{\alpha}, \frac{\beta-\gamma}{\alpha}, \frac{3+\alpha-\gamma}{\alpha}, -x_c(r)^\alpha \right) (3-\gamma)^{-1},$$

where,

$$\alpha = 2.94 - \log_{10} \left[\left(10^{X+2.33} \right)^{-1.08} + \left(10^{X+2.33} \right)^{2.99} \right],$$

$$\beta = 4.23 + 1.34X + 0.26X^2,$$

$$\gamma = -0.06 - \log_{10} \left[\left(10^{X+2.56} \right)^{-0.68} + 10^{X+2.56} \right].$$

Here, $X = \text{cdm_funcs.base_funcs.mass_frac_dc14}$, and ${}_2F_1 =$ Gaussian hypergeometric function.

- Einasto,

$$M_{\text{Einasto}}(r) = 4\pi \rho_c r_c^3 \exp(2/\alpha) (2/\alpha)^{-3/\alpha} \Gamma(3/\alpha, 2x_c^\alpha/\alpha) / \alpha,$$

where $\alpha =$ alpha in [cdm_params.halo.params](#) and $\Gamma(a, x) =$ incomplete Gamma function.

- NFW,

$$M_{\text{NFW}}(r) = 4\pi \rho_c r_c^3 \left[\ln(1 + x_c(r)) - \frac{x_c(r)}{1 + x_c(r)} \right].$$

For each of the above masses, $\rho_c = \text{cdm_funcs.base_funcs.rhoc}$, $r_c = \text{cdm_funcs.base_funcs.rc}$, and $x_c(r) = \text{cdm_funcs.base_funcs.xc}$.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N]
Numpy array of N mass values in units of solar mass.

5.7.3.2 Mvir()

```
def cdm_funcs.halo.Mvir (
    self,
    params )
```

Define the total galactic DM halo mass.

This defines the total mass of the galactic DM given the model parameters. The total galactic DM mass is given by,

$$M_{200} = \sqrt{\frac{3}{2\pi\rho_{\text{crit}}}} \frac{M_P^3 V_{200}^3}{20},$$

where $\rho_{\text{crit}} = \text{rhocrit}$ and $M_P = \text{MP}$ in [constants.standard](#), while $V_{200} = \text{v200}$ in [cdm_params.halo.params](#).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .

Returns

float
Total mass of the galactic DM in units of solar mass.

5.7.4 Member Data Documentation**5.7.4.1 mass_num**

`cdm_funcs.halo.mass_num`

int
Used to differentiate between soliton 1 and soliton 2 in double flavored ULDM models.

Outer CDM halo is halo 1 if `mass_num = 0` and is halo 2 if `mass_num = 1`.

The documentation for this class was generated from the following file:

- `pyfiles/models/cdm/cdm_funcs.py`

5.8 cdm_params.halo Class Reference

The class containing all parameters for fitting the CDM halo models.

Public Member Functions

- `def __init__(self, model, data, fit_dict_in=fitting_dict_in)`
Define the constructor of the halo class.
- `def params(self)`
Define the parameters to be assumed.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **data**
dictionary
Dictionary containing all the necessary data for a given galaxy.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the `constants.fitting_dict` class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of `constants.fitting_dict.args_opts`
- **params_vals**

- dictionary*
Dictionary of variable values to be used during fitting.
- **c200**
list
Values for c200 in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
 - **v200**
list
Values for V200 in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
 - **v200fac**
list
Values for v200fac in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
 - **MLD**
list
Values for MLD in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
 - **MLB**
list
Values for MLB in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
 - **alpha**
list
Values for alpha in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)

5.8.1 Detailed Description

The class containing all parameters for fitting the CDM halo models.

This class contains all parameters necessary to perform the fitting procedures for all CDM halo models.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 __init__()

```
def cdm_params.halo.__init__ (
    self,
    model,
    data,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the halo class.

This defines the constructor of the halo class.

Parameters

<i>self</i>	object pointer
-------------	----------------

Parameters

<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none">• Burkert• DC14• Einasto• NFW
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .
<i>fit_dict←_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.8.3 Member Function Documentation

5.8.3.1 params()

```
def cdm_params.halo.params (
    self )
```

Define the parameters to be assumed.

This defines the parameters to be assumed during the fitting procedures for all CDM halos.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

Imfit.Parameters instance Instance of the Imfit.Parameters class

5.8.4 Member Data Documentation

5.8.4.1 data

```
cdm_params.halo.data
```

dictionary

Dictionary containing all the necessary data for a given galaxy.

This can be created using [galaxy.galaxy](#).

5.8.4.2 params_vals

`cdm_params.halo.params_vals`

dictionary

Dictionary of variable values to be used during fitting.

Instance of [constants.fitting_dict.params_vals](#)

The documentation for this class was generated from the following file:

- `pyfiles/models/cdm/cdm_params.py`

5.9 halo.halo Class Reference

The class containing definitions to describe a DM halo.

Public Member Functions

- `def __init__(self, model, data, fit_dict_in=fitting_dict_in)`
Define the constructor of the halo class.
- `def mass(self, params, r)`
Define the DM halo mass within some radius assuming the chosen model.
- `def velocity(self, params, r)`
Define the total circular velocity of the DM halo at some radius assuming the chosen model.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **data**
dictionary
Dictionary containing all the necessary data for a given galaxy.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the constants.fitting_dict class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of constants.fitting_dict.args_opts
- **params_vals**
dictionary
Dictionary of variable values to be used during fitting.
- **params**
Imfit.Parameters instance
Instance of the Imfit.Parameters class All model parameters contained here.
- **halo_init**
Can be :

5.9.1 Detailed Description

The class containing definitions to describe a DM halo.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 `__init__()`

```
def halo.halo.__init__ (
    self,
    model,
    data,
    fit_dict_in = fitting\_dict\_in )
```

Define the constructor of the halo class.

This defines the constructor of the halo class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert • DC14 • Einasto • NFW • psi_single • psi_multi
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.9.3 Member Function Documentation

5.9.3.1 `mass()`

```
def halo.halo.mass (
    self,
```

```

        params,
        r )

```

Define the DM halo mass within some radius assuming the chosen model.

This defines the total DM halo mass within the given radius assuming the given model parameters. Particular equations for mass depends on chosen model.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N]
Numpy array of N mass values in units of solar mass.

5.9.3.2 velocity()

```

def halo.halo.velocity (
    self,
    params,
    r )

```

Define the total circular velocity of the DM halo at some radius assuming the chosen model.

This calculates the total circular velocity of the DM halo at the given radius assuming the given model parameters. Total circular velocity is given by,

$$V(r) = \sqrt{\frac{M(r)}{M_P^2 r}}$$

where $M(r)$ = [halo.halo.mass](#), and M_P is the Planck mass (see [constants.standard](#))

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N]
Numpy array of N circular velocity values in units of km/s.

5.9.4 Member Data Documentation

5.9.4.1 data

`halo.halo.data`

dictionary
Dictionary containing all the necessary data for a given galaxy.

This can be created using [galaxy.galaxy](#).

5.9.4.2 halo_init

`halo.halo.halo_init`

Can be :

- [cdm_funcs.halo](#) instance
- [alp_funcs.soliton](#) instance
Instance of [cdm_funcs.halo](#) or [alp_funcs.soliton](#) class depending on chosen model.

5.9.4.3 params

`halo.halo.params`

Imfit.Parameters instance
Instance of the Imfit.Parameters class All model parameters contained here.

This can be created using [cdm_params.halo.params](#) (for CDM models) or [alp_params.soliton.params](#) (for ULDM models)

5.9.4.4 params_vals

`halo.halo.params_vals`

dictionary
Dictionary of variable values to be used during fitting.

Instance of [constants.fitting_dict.params_vals](#)

The documentation for this class was generated from the following file:

- `pyfiles/data_models/halo.py`

5.10 model_fit.model_fit Class Reference

The class containing the fitting procedure assuming a chosen halo model for a chosen galaxy.

Public Member Functions

- `def __init__ (self, halo, ULDM_fits=False)`
Define the constructor of the [model_fit](#) class.
- `def velocity_tot (self, params, r, data)`
Define the total circular velocity of a galaxy at some radius for the given halo model.
- `def residual (self, params, r, data)`
Define the residual of the modeled total circular velocity from the data for a chosen galaxy.
- `def fit (self, params, data, calc_covar_in=True)`
Define the fit result for the modeled total circular velocity for a galaxy.
- `def bic (self, fit)`
Define the BIC value for a given fit.
- `def plot (self, halo, gal, axs)`
Define the plot of the model of the total circular velocity for a galaxy.

Public Attributes

- **halo**
[halo.halo](#) instance
Instance of the [halo.halo](#) class.
- **args_opts**
 dictionary
Dictionary of rules to be assumed during fitting.
- **ULDM_fits**
 bool
True if performing fits for the ULDM models

5.10.1 Detailed Description

The class containing the fitting procedure assuming a chosen halo model for a chosen galaxy.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 __init__()

```
def model_fit.model_fit.__init__ (
    self,
    halo,
    ULDM_fits = False )
```

Define the constructor of the [model_fit](#) class.

This defines the constructor of the [model_fit](#) class.

Parameters

<i>self</i>	object pointer
<i>halo</i>	halo.halo instance Instance of the halo.halo class. Assumed halo model and proper definitions contained in halo class instance.
<i>ULDM_fits</i>	(optional) bool True if performing fits for the ULDM models

5.10.3 Member Function Documentation

5.10.3.1 `bic()`

```
def model_fit.model_fit.bic (
    self,
    fit )
```

Define the BIC value for a given fit.

This defines the BIC value for a given fit.

Parameters

<i>self</i>	object pointer
<i>fit</i>	model_fit.model_fit.fit instance Instance of the model_fit.model_fit.fit function.

Returns

float
BIC value for a given fit.

5.10.3.2 `fit()`

```
def model_fit.model_fit.fit (
    self,
    params,
    data,
    calc_covar_in = True )
```

Define the fit result for the modeled total circular velocity for a galaxy.

This defines the fit result for the modeled total circular velocity assuming the given model parameters and galaxy data.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .
<i>calc_covar↔ _in</i>	(optional) bool True(False) if covariance matrix is to be(is not to be) calculated in fitting procedure

Returns

Imfit.Minimizer.minimize fit result

Fit result contains the optimized parameters and several goodness-of-fit statistics.

5.10.3.3 plot()

```
def model_fit.model_fit.plot (
    self,
    halo,
    gal,
    axs )
```

Define the plot of the model of the total circular velocity for a galaxy.

This defines the plot of the model of total circular velocity for a galaxy assuming the best fit parameters.

Parameters

<i>self</i>	object pointer
<i>halo</i>	halo.halo instance Instance of the halo.halo class. Assumed halo model and proper definitions contained in halo class instance.
<i>gal</i>	galaxy.galaxy instance Instance of the galaxy.galaxy class. Contains all necessary galaxy data.
<i>axs</i>	matplotlib axis Axis in which matplotlib plot will be contained

Returns

matplotlib plot

Plot of the circular velocity of the different galaxy contributions including the DM halo given the assumed model and best fit parameters.

5.10.3.4 residual()

```
def model_fit.model_fit.residual (
    self,
    params,
    r,
    data )
```

Define the residual of the modeled total circular velocity from the data for a chosen galaxy.

This calculates the residual of the modeled total circular velocity from the data for a galaxy at a given radius assuming the given model parameters and galaxy data. This is used as the fitting residual in the `Imfit.Minimizer` class and is given by,

$$\text{res} = \frac{V_{\text{obs}}(r) - V_{\text{model}}(r)}{V_{\text{obs,err}}(r)},$$

where $V_{\text{obs}}(r)$ and $V_{\text{obs,err}}(r)$ can be found in [galaxy.galaxy.data](#) and $V_{\text{model}}(r) = \text{model_fit.model_fit.velocity_tot}$.

Parameters

<i>self</i>	object pointer
<i>params</i>	<code>Imfit.Parameters</code> instance Instance of the <code>Imfit.Parameters</code> class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>r</i>	<code>ndarray[N]</code> Numpy array of N radii values in units of kpc.
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .

Returns

`ndarray[N]`
Numpy array of N residual values.

5.10.3.5 velocity_tot()

```
def model_fit.model_fit.velocity_tot (
    self,
    params,
    r,
    data )
```

Define the total circular velocity of a galaxy at some radius for the given halo model.

This defines the total circular velocity of a galaxy at a given radius assuming the given model parameters and galaxy data. The total circular velocity is given by,

$$V(r) = \sqrt{V_{\text{bar}}(r)^2 + V_{\text{DM}}(r)^2},$$

where,

$$V_{\text{bar}}(r) = \sqrt{|V_{\text{gas}}(r)| V_{\text{gas}}(r) + \tilde{\Upsilon}_{\text{disk}} |V_{\text{disk}}(r)| V_{\text{disk}}(r) + \tilde{\Upsilon}_{\text{bulge}} |V_{\text{bulge}}(r)| V_{\text{bulge}}(r)}.$$

Here, $V_{\text{DM}}(r) = \text{halo.halo.velocity}$, $V_{\text{gas}}(r)$, $V_{\text{disk}}(r)$, and $V_{\text{bulge}}(r)$ can be found in [galaxy.galaxy.data](#). Finally, $\tilde{\Upsilon}_{\text{disk}} = \text{MLD}$ and $\tilde{\Upsilon}_{\text{bulge}} = \text{MLB}$ in [cdm_params.halo.params](#) (for CDM models) or [alp_params.soliton.params](#) (for ULDM models).

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params (for CDM models) or alp_params.soliton.params (for ULDM models)
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .

Returns

ndarray[N]
Numpy array of N circular velocity values in units of km/s.

5.10.4 Member Data Documentation

5.10.4.1 args_opts

`model_fit.model_fit.args_opts`

dictionary
Dictionary of rules to be assumed during fitting.

Instance of [constants.fitting_dict.args_opts](#)

The documentation for this class was generated from the following file:

- `pyfiles/fitting/model_fit.py`

5.11 results.plots Class Reference

The class to obtain rotation curve plots for given galaxies.

Public Member Functions

- def `__init__` (self)
Define the constructor of the plots class.
- def `rotcurves_CDM_all` (self, galaxies, `fit_dict_in=fitting_dict_in`, `size=(20, 20)`, `save_file=None`)
Define the plot of the rotation curves for given galaxies and for all CDM models.
- def `rotcurves_psi_all` (self, galaxies, `fit_dict_in=fitting_dict_in`, `size=(20, 20)`, `save_file=None`)
Define the plot of the rotation curves for given galaxies and for the single ULDM and double ULDM models.
- def `rotcurves_CDM_check` (self, galaxies, `fit_dict_in=fitting_dict_in`, `size=(20, 20)`)
Define the plot of the rotation curves for given galaxies and for all CDM models.
- def `rotcurves_DC14_check` (self, galaxies, `fit_dict_in=fitting_dict_in`, `size=(20, 20)`)
Define the plot of the rotation curves for given galaxies and for the DC14 and NFW models.
- def `rotcurves_Einasto_check` (self, galaxies, `fit_dict_in=fitting_dict_in`, `size=(20, 20)`)
Define the plot of the rotation curves for given galaxies and for the Einasto and NFW models.
- def `BIC_diffs_CDM` (self, dict_in, bins=250, lwin=3, `save_file=None`)
Define the plot of the BIC differences for different priors using CDM models.
- def `BIC_CDM` (self, dict_in, bins=250, lwin=3, `save_file=None`)
Define the plot of the BIC differences for the CDM models.
- def `chi_CDM` (self, dict_in, spl=50, lwin=3, `save_file=None`)
Define the plot of the reduced chi-square comparisons for the CDM models.
- def `chi_dist_CDM` (self, dict_in, bins=50, `save_file=None`)
Define the plot of the reduced chi-square distributions for the CDM models.
- def `params_dist_CDM` (self, dict_in, bins=50, `save_file=None`)
Define the plot of the parameter distributions for the CDM models.
- def `MLd_degen_CDM` (self, `save_file=None`)
Define the plot of the reduced chi-square contours for the NFW model.
- def `relations_CDM` (self, dict_in, `save_file=None`)
Define the plot of the empirical relations for the CDM models.
- def `BIC_psi_mfree` (self, psis_dict_in, psim_dict_in, CDM_dict_in, spl=50, lwin=3, `save_file=None`)
Define the plot of the BIC differences between the ULDM (particle mass free) and Einasto models.
- def `chi_psi_mfree` (self, psis_dict_in, psim_dict_in, CDM_dict_in, spl=50, lwin=3, `save_file=None`)
Define the plot of the reduced chi-square for the ULDM (particle mass free) and Einasto models.
- def `Msol_psi_mfree` (self, psis_dict_in, psim_dict_in, `save_file=None`)
Define the plot of the soliton-halo (SH) relation for the ULDM (particle mass free) models.
- def `chi_dist_psi_mfree` (self, psis_dict_in, psim_dict_in, bins=50, `save_file=None`)
Define the plot of the reduced chi-square distributions for the ULDM (particle mass free) models.
- def `params_dist_psi_mfree` (self, psis_dict_in, psim_dict_in, bins=50, `save_file=None`)
Define the plot of the parameter distributions for the ULDM (particle mass free) models.
- def `relations_psi_mfree` (self, psis_dict_in, psim_dict_in, `save_file=None`)
Define the plot of the empirical relations for the ULDM (particle mass free) models.
- def `chi_psi_mfix` (self, psis_dict_in, psim_dict_in, CDM_dict_in, lwin=4, `fit_dict_in=fitting_dict_in`, `save_file=None`)
Define the plot of the reduced chi-square for the ULDM (particle mass fixed and scanned) and Einasto models.
- def `chi_gal_psi_mfix` (self, psis_dict_in, CDM_dict_in, lwin=4, `fit_dict_in=fitting_dict_in`, `save_file=None`)
Define the plot of the cumulative reduced chi-square for single flavor ULDM (particle mass fixed and scanned) and Einasto models.
- def `Msol_psi_mfix` (self, psis_dict_in, psim_dict_in, bins=50, `fit_dict_in=fitting_dict_in`, `save_file=None`)
Define the plot of the soliton-halo (SH) relation for the ULDM (particle mass fixed and scanned) models.
- def `BIC_psi_mfix_ex` (self, psis_dict_in, psim_dict_in, CDM_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, bins=50, lwin=3, `save_file=None`)
Define the plot of the BIC differences distributions between the ULDM (particle mass fixed) and Einasto models.

- def [chi_psi_mfix_ex](#) (self, psis_dict_in, psim_dict_in, CDM_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, spl=50, lwin=3, save_file=None)
Define the plot of the reduced chi-square for the ULDM (particle mass fixed) and Einasto models.
- def [Msol_psi_mfix_ex](#) (self, psis_dict_in, psim_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, save_file=None)
Define the plot of the soliton-halo (SH) for the ULDM (particle mass fixed) and Einasto models.
- def [chi_dist_psi_mfix_ex](#) (self, psis_dict_in, psim_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, bins=50, save_file=None)
Define the plot of the reduced chi-square distributions for the ULDM (particle mass fixed) models.
- def [params_dist_psi_mfix_ex](#) (self, psis_dict_in, psim_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, bins=50, save_file=None)
Define the plot of the parameter distributions for the ULDM (particle mass fixed) models.
- def [relations_psi_mfix_ex](#) (self, psis_dict_in, psim_dict_in, m22_ex=10 $^{**}(1.5)$, m22_2_ex=10 $^{**}(1.8)$, save_file=None)
Define the plot of the empirical relations for the ULDM (particle mass fixed) models.
- def [chi_dist_CDM_checks](#) (self, dict_in, bins=50)
Define the plot of the reduced chi-square distributions for the CDM models (checks).
- def [params_dist_CDM_checks](#) (self, dict_in, bins=50)
Define the plot of the parameter distributions for the CDM models (checks).
- def [chi_dist_DC14_checks](#) (self, dict_in, bins=50)
Define the plot of the reduced chi-square distributions for the DC14 and NFW models (checks).
- def [params_dist_DC14_checks](#) (self, dict_in, bins=50)
Define the plot of the parameter distributions for the DC14 and NFW models (checks).
- def [chi_box_Einasto_checks](#) (self, dict_in)
Define the plot of the reduced chi-square for the Einasto and NFW models (checks).
- def [chi_dist_Einasto_checks](#) (self, dict_in, bins=50)
Define the plot of the reduced chi-square distributions for the Einasto and NFW models (checks).
- def [params_dist_Einasto_checks](#) (self, dict_in, bins=50)
Define the plot of the parameters distributions for the Einasto and NFW models (checks).
- def [params_scatter_Einasto_checks](#) (self, dict_in, fit_dict_in=fitting_dict_in)
Define the plot of the halo parameters for the Einasto model (checks).

Public Attributes

- [fit_dict_in](#)
constants.fitting_dict instance
Instance of the constants.fitting_dict class.
- [size](#)
tuple
Size of figure.

5.11.1 Detailed Description

The class to obtain rotation curve plots for given galaxies.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 `__init__()`

```
def results.plots.__init__ (
    self )
```

Define the constructor of the plots class.

This defines the constructor of the plots class.

Parameters

<i>self</i>	object pointer
<i>galaxies</i>	str[N] List of galaxy names (or 'random') to plot the given galaxies (or random galaxies) in the SPARC catalog.

5.11.3 Member Function Documentation

5.11.3.1 `BIC_CDM()`

```
def results.plots.BIC_CDM (
    self,
    dict_in,
    bins = 250,
    lwin = 3,
    save_file = None )
```

Define the plot of the BIC differences for the CDM models.

This defines the plot of the differences in BIC between each CDM model analyzed.

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See <code>fits_CDM_all.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,3]
Figure of the differences in BIC between each CDM model.

5.11.3.2 BIC_diffs_CDM()

```
def results.plots.BIC_diffs_CDM (
    self,
    dict_in,
    bins = 250,
    lwin = 3,
    save_file = None )
```

Define the plot of the BIC differences for different priors using CDM models.

This defines the plot of the differences in BIC for different prior cases for each CDM model analyzed.

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See fits_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'uni_priors' : {...} • 'c200_priors_check' : {...} • 'v200_priors_check' : {...} • 'MLd_priors_check' : {...} • 'MLb_priors_check' : {...} }, where each {...} must be of the form: {...} = • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} Finally, each {...} must be an instance of results.results_CDM_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the BIC differences for different priors using all CDM models.

5.11.3.3 BIC_psi_mfix_ex()

```
def results.plots.BIC_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
    CDM_dict_in,
    m22_ex = 10*(1.5),
    m22_2_ex = 10*(1.8),
    bins = 50,
    lwin = 3,
    save_file = None )
```

Define the plot of the BIC differences distributions between the ULDM (particle mass fixed) and Einasto models.

This defines the plot of the distributions of the differences in BIC between each of the ULDM (particle mass fixed) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>lwin</i>	(optional) int Width of lines for plot.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the BIC differences distributions between each ULDM (particle mass fixed) model and the Einasto model.

5.11.3.4 BIC_psi_mfree()

```
def results.plots.BIC_psi_mfree (
    self,
    psis_dict_in,
    psim_dict_in,
    CDM_dict_in,
    splt = 50,
    lwin = 3,
    save_file = None )
```

Define the plot of the BIC differences between the ULDM (particle mass free) and Einasto models.

This defines the plot of the differences in BIC vs. ULDM particle mass between each of the ULDM (particle mass free) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>splt</i>	(optional) int Point size for plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[3,2]

Figure of the differences in BIC vs. particle mass between each ULDM (particle mass free) model and the Einasto model.

5.11.3.5 chi_box_Einasto_checks()

```
def results.plots.chi_box_Einasto_checks (
    self,
    dict_in )
```

Define the plot of the reduced chi-square for the Einasto and NFW models (checks).

This defines the plot of the reduced chi-square comparison between the Einasto and NFW models (checks).

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See checks_Einasto_NFW.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_Einasto_check.fit.

Returns

matplotlib.pyplot.figure[1,2]

Figure of the reduced chi-square comparisons between the Einasto and NFW models (checks).

5.11.3.6 chi_CDM()

```
def results.plots.chi_CDM (
    self,
    dict_in,
    splt = 50,
    lwin = 3,
    save_file = None )
```

Define the plot of the reduced chi-square comparisons for the CDM models.

This defines the plot of the reduced chi-square comparisons between each CDM model analyzed.

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See fits_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_all.fit.

Parameters

<i>splt</i>	(optional) int Point size for plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,3]

Figure of the reduced chi-square comparisons between each CDM model.

5.11.3.7 chi_dist_CDM()

```
def results.plots.chi_dist_CDM (
    self,
    dict_in,
    bins = 50,
    save_file = None )
```

Define the plot of the reduced chi-square distributions for the CDM models.

This defines the plot of the reduced chi-square distribution for each CDM model analyzed.

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See fits_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square distributions for each CDM model.

5.11.3.8 chi_dist_CDM_checks()

```
def results.plots.chi_dist_CDM_checks (
    self,
    dict_in,
    bins = 50 )
```

Define the plot of the reduced chi-square distributions for the CDM models (checks).

This defines the plot of the reduced chi-square distribution for each CDM model analyzed (checks).

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See checks_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[1,4]

Figure of the reduced chi-square distributions for each CDM model (checks).

5.11.3.9 chi_dist_DC14_checks()

```
def results.plots.chi_dist_DC14_checks (
    self,
    dict_in,
    bins = 50 )
```

Define the plot of the reduced chi-square distributions for the DC14 and NFW models (checks).

This defines the plot of the reduced chi-square distribution for the DC14 and NFW models (checks).

Parameters

<i>self</i>	object pointer
-------------	----------------

Parameters

<i>dict_in</i>	dictionary See checks_DC14_NFW.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'DC14' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_DC14_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[1,2]

Figure of the reduced chi-square distributions for the DC14 and NFW models (checks).

5.11.3.10 chi_dist_Einasto_checks()

```
def results.plots.chi_dist_Einasto_checks (
    self,
    dict_in,
    bins = 50 )
```

Define the plot of the reduced chi-square distributions for the Einasto and NFW models (checks).

This defines the plot of the reduced chi-square distribution for the Einasto and NFW models (checks).

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See checks_Einasto_NFW.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_Einasto_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[1,2]

Figure of the reduced chi-square distributions for the Einasto and NFW models (checks).

5.11.3.11 `chi_dist_psi_mfix_ex()`

```
def results.plots.chi_dist_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
    m22_ex = 10*(1.5),
    m22_2_ex = 10*(1.8),
    bins = 50,
    save_file = None )
```

Define the plot of the reduced chi-square distributions for the ULDM (particle mass fixed) models.

This defines the plot of the reduced chi-square distributions for each of the ULDM (particle mass fixed) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See <code>fits_Einasto_all.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: <code>psis_dict_in = {</code> <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See <code>fits_Einasto_all.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: <code>psim_dict_in = {</code> <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

`matplotlib.pyplot.figure[2,2]`

Figure of the reduced chi-square distributions for each of the ULDM (particle mass fixed) models.

5.11.3.12 `chi_dist_psi_mfree()`

```
def results.plots.chi_dist_psi_mfree (
    self,
```

```

    psis_dict_in,
    psim_dict_in,
    bins = 50,
    save_file = None )

```

Define the plot of the reduced chi-square distributions for the ULDM (particle mass free) models.

This defines the plot of the reduced chi-square distributions for each of the ULDM (particle mass free) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square distributions for each of the ULDM (particle mass free) models.

5.11.3.13 chi_gal_psi_mfix()

```

def results.plots.chi_gal_psi_mfix (
    self,
    psis_dict_in,
    CDM_dict_in,
    lwin = 4,
    fit_dict_in = fitting_dict_in,
    save_file = None )

```

Define the plot of the cumulative reduced chi-square for single flavor ULDM (particle mass fixed and scanned) and Einasto models.

This defines the plot of the cumulative reduced chi-square comparisons vs. galaxy between each of the single flavor ULDM (particle mass fixed and scanned) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>lwin</i>	(optional) int Width of lines for plot.
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains the mass ranges to be used.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,1]

Figure of the cumulative reduced chi-square vs. galaxy for each of the single flavor ULDM (particle mass fixed and scanned) models and the Einasto model.

5.11.3.14 chi_psi_mfix()

```
def results.plots.chi_psi_mfix (
    self,
    psis_dict_in,
    psim_dict_in,
    CDM_dict_in,
    lwin = 4,
    fit_dict_in = fitting_dict_in,
    save_file = None )
```

Define the plot of the reduced chi-square for the ULDM (particle mass fixed and scanned) and Einasto models.

This defines the plot of the reduced chi-square comparisons vs. particle mass between each of the ULDM (particle mass fixed and scanned) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>lwin</i>	(optional) int Width of lines for plot.
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains the mass ranges to be used.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square comparisons vs. particle mass between each of the ULDM (particle mass fixed and scanned) models and the Einasto model.

5.11.3.15 chi_psi_mfix_ex()

```
def results.plots.chi_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
    CDM_dict_in,
    m22_ex = 10**(1.5),
    m22_2_ex = 10**(1.8),
    splt = 50,
    lwin = 3,
    save_file = None )
```

Define the plot of the reduced chi-square for the ULDM (particle mass fixed) and Einasto models.

This defines the plot of the reduced chi-square comparisons between each of the ULDM (particle mass fixed) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>splt</i>	(optional) int Point size for plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square comparisons between each ULDM (particle mass fixed) model and the Einasto model.

5.11.3.16 chi_psi_mfree()

```
def results.plots.chi_psi_mfree (
    self,
    psis_dict_in,
    psim_dict_in,
    CDM_dict_in,
    splt = 50,
    lwin = 3,
    save_file = None )
```

Define the plot of the reduced chi-square for the ULDM (particle mass free) and Einasto models.

This defines the plot of the reduced chi-square comparisons between each of the ULDM (particle mass free) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>CDM_dict_in</i>	dictionary The dictionary for the Einasto. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be an instance of results.results_CDM_all.fit .
<i>splt</i>	(optional) int Point size for plot.
<i>lwin</i>	(optional) int Width of lines for plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square comparisons between each of the ULDM (particle mass free) models and the Einasto model.

5.11.3.17 MLd_degen_CDM()

```
def results.plots.MLd_degen_CDM (
    self,
    save_file = None )
```

Define the plot of the reduced chi-square contours for the NFW model.

This defines the plot of the reduced chi-square contours in the $\tilde{\Upsilon}_{d-c_{200}}$ plane for the NFW model.

Parameters

<i>self</i>	object pointer
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[2,2]

Figure of the reduced chi-square contours in the $\tilde{\Upsilon}_{\text{d-c}_{200}}$ plane for the NFW model.

5.11.3.18 Msol_psi_mfix()

```
def results.plots.Msol_psi_mfix (
    self,
    psis_dict_in,
    psim_dict_in,
    bins = 50,
    fit_dict_in = fitting_dict_in,
    save_file = None )
```

Define the plot of the soliton-halo (SH) relation for the ULDM (particle mass fixed and scanned) models.

This defines the plot of the SH relation for each of the ULDM (particle mass fixed and scanned) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains the mass ranges to be used.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[3,2]

Figure of the SH relation for each of the ULDM (particle mass fixed and scanned) models.

5.11.3.19 Msol_psi_mfix_ex()

```
def results.plots.Msol_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
    m22_ex = 10*(1.5),
    m22_2_ex = 10*(1.8),
    save_file = None )
```

Define the plot of the soliton-halo (SH) for the ULDM (particle mass fixed) and Einasto models.

This defines the plot of the SH relation for each of the ULDM (particle mass fixed) models and the Einasto model.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[3,2]

Figure of the SH relation for each ULDM (particle mass fixed) model and the Einasto model.

5.11.3.20 Msol_psi_mfree()

```
def results.plots.Msol_psi_mfree (
    self,
    psis_dict_in,
```

```
psim_dict_in,  
save_file = None )
```

Define the plot of the soliton-halo (SH) relation for the ULDM (particle mass free) models.

This defines the plot of the SH relation vs. particle mass for each of the ULDM (particle mass free) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[3,2]

Figure of the SH relation vs. particle mass for each of the ULDM (particle mass free) models.

5.11.3.21 params_dist_CDM()

```
def results.plots.params_dist_CDM (
    self,
    dict_in,
    bins = 50,
    save_file = None )
```

Define the plot of the parameter distributions for the CDM models.

This defines the plot of the parameter distributions for each CDM model analyzed.

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See fits_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the parameter distributions for each CDM model.

5.11.3.22 params_dist_CDM_checks()

```
def results.plots.params_dist_CDM_checks (
    self,
    dict_in,
    bins = 50 )
```

Define the plot of the parameter distributions for the CDM models (checks).

This defines the plot of the parameter distributions for each CDM model analyzed (checks).

Parameters

<i>self</i>	object pointer
<i>dict_in</i>	dictionary See checks_CDM_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Burkert' : {...} • 'DC14' : {...} • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_CDM_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the parameter distributions for each CDM model (checks).

5.11.3.23 params_dist_DC14_checks()

```
def results.plots.params_dist_DC14_checks (
    self,
    dict_in,
    bins = 50 )
```

Define the plot of the parameter distributions for the DC14 and NFW models (checks).

This defines the plot of the parameter distributions for the DC14 and NFW models (checks).

Parameters

<i>self</i>	object pointer
<i>dict↔ _in</i>	dictionary See checks_DC14_NFW.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'DC14' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_DC14_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[1,2]
Figure of the reduced chi-square distributions for the DC14 and NFW models (checks).

5.11.3.24 params_dist_Einasto_checks()

```
def results.plots.params_dist_Einasto_checks (
    self,
    dict↔_in,
    bins = 50 )
```

Define the plot of the parameters distributions for the Einasto and NFW models (checks).

This defines the plot of the parameter distributions for the Einasto and NFW models (checks).

Parameters

<i>self</i>	object pointer
<i>dict↔ _in</i>	dictionary See checks_Einasto_NFW.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_Einasto_check.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.

Returns

matplotlib.pyplot.figure[3,2]
Figure of the parameter distributions for the Einasto and NFW models (checks).

5.11.3.25 params_dist_psi_mfix_ex()

```
def results.plots.params_dist_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
    m22_ex = 10**(1.5),
    m22_2_ex = 10**(1.8),
    bins = 50,
    save_file = None )
```

Define the plot of the parameter distributions for the ULDM (particle mass fixed) models.

This defines the plot of the parameter distributions for each of the ULDM (particle mass fixed) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the parameter distributions for each of the ULDM (particle mass fixed) models.

5.11.3.26 params_dist_psi_mfree()

```
def results.plots.params_dist_psi_mfree (
    self,
```



```

    psis_dict_in,
    psim_dict_in,
    bins = 50,
    save_file = None )

```

Define the plot of the parameter distributions for the ULDM (particle mass free) models.

This defines the plot of the parameter distributions for each of the ULDM (particle mass free) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>bins</i>	(optional) int Number of bins for histogram plot.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]

Figure of the parameter distributions for each of the ULDM (particle mass free) models.

5.11.3.27 params_scatter_Einasto_checks()

```

def results.plots.params_scatter_Einasto_checks (
    self,
    dict_in,
    fit_dict_in = fitting_dict_in )

```

Define the plot of the halo parameters for the Einasto model (checks).

This defines the plot of the halo parameters for the Einasto model (checks).

Parameters

<i>self</i>	object pointer
-------------	----------------

Parameters

<i>dict_in</i>	dictionary See <code>checks_Einasto_NFW.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: dict_in = { <ul style="list-style-type: none"> • 'Einasto' : {...} • 'NFW' : {...} }, where each {...} must be an instance of results.results_Einasto_check.fit.
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains the necessary parameters for fitting.

Returns

matplotlib.pyplot.figure[1,2]
Figure of the halo parameters for the Einasto model (checks).

5.11.3.28 relations_CDM()

```
def results.plots.relations_CDM (
    self,
    dict_in,
    save_file = None )
```

Define the plot of the empirical relations for the CDM models.

This defines the plot of the empirical relations for each CDM models analyzed.

Parameters

<i>self</i>	object pointer
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the empirical relations for each CDM model.

5.11.3.29 relations_psi_mfix_ex()

```
def results.plots.relations_psi_mfix_ex (
    self,
    psis_dict_in,
    psim_dict_in,
```

```

m22_ex = 10**(1.5),
m22_2_ex = 10**(1.8),
save_file = None )

```

Define the plot of the empirical relations for the ULDM (particle mass fixed) models.

This defines the plot of the empirical relations for each of the ULDM (particle mass fixed) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psis_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See fits_Einasto_all.ipynb for an example of formatting for the dictionary. Dictionary must be of the form: psim_dict_in = { <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>m22_ex</i>	(optional) float Value for particle mass (single flavor models) or particle mass one (double flavor models).
<i>m22_2_ex</i>	(optional) float Value for particle mass two (double flavor models).
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[4,4]
Figure of the empirical relations for each of the ULDM (particle mass fixed) models.

5.11.3.30 relations_psi_mfree()

```

def results.plots.relations_psi_mfree (
    self,
    psis_dict_in,
    psim_dict_in,
    save_file = None )

```

Define the plot of the empirical relations for the ULDM (particle mass free) models.

This defines the plot of the empirical relations for each of the ULDM (particle mass free) models.

Parameters

<i>self</i>	object pointer
<i>psis_dict_in</i>	dictionary The dictionary for the single flavor models. See <code>fits_Einasto_all.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: <code>psis_dict_in = {</code> <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_single_all.fit.
<i>psim_dict_in</i>	dictionary The dictionary for the double flavor models. See <code>fits_Einasto_all.ipynb</code> for an example of formatting for the dictionary. Dictionary must be of the form: <code>psim_dict_in = {</code> <ul style="list-style-type: none"> • 'Summed' : {...} • 'Matched' : {...} }, where each {...} must be an instance of results.results_psi_multi_all.fit.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

`matplotlib.pyplot.figure[4,4]`

Figure of the empirical relations for each of the ULDM (particle mass free) models.

5.11.3.31 `rotcurves_CDM_all()`

```
def results.plots.rotcurves_CDM_all (
    self,
    galaxies,
    fit_dict_in = fitting_dict_in,
    size = (20,20),
    save_file = None )
```

Define the plot of the rotation curves for given galaxies and for all CDM models.

This defines the plot of the rotation curves for all given galaxies and for all CDM models.

Parameters

<i>self</i>	object pointer
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.
<i>size</i>	(optional) tuple Size of figure.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[N,4]

Figure of rotation curves for all given galaxies (N galaxies in total) and for all CDM halos (4 in total).

5.11.3.32 rotcurves_CDM_check()

```
def results.plots.rotcurves_CDM_check (
    self,
    galaxies,
    fit_dict_in = fitting_dict_in,
    size = (20,20) )
```

Define the plot of the rotation curves for given galaxies and for all CDM models.

This defines the plot of the rotation curves for all given galaxies and for all CDM models. For comparison with Pengfei Li et al 2020 ApJS 247 31 : <https://doi.org/10.3847/1538-4365/ab700e>

Parameters

<i>self</i>	object pointer
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.
<i>size</i>	(optional) tuple Size of figure.

Returns

matplotlib.pyplot.figure[N,4]

Figure of rotation curves for all given galaxies (N galaxies in total) and for all CDM halos (4 in total).

5.11.3.33 rotcurves_DC14_check()

```
def results.plots.rotcurves_DC14_check (
    self,
    galaxies,
    fit_dict_in = fitting_dict_in,
    size = (20,20) )
```

Define the plot of the rotation curves for given galaxies and for the DC14 and NFW models.

This defines the plot of the rotation curves for all given galaxies and for the DC14 and NFW models. For comparison with Monthly Notices of the Royal Astronomical Society, Volume 466, Issue 2, April 2017, Pages 1648–1668 : <https://doi.org/10.1093/mnras/stw3101>

Parameters

<i>self</i>	object pointer
<i>fit_dict↔ _in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.
<i>size</i>	(optional) tuple Size of figure.

Returns

matplotlib.pyplot.figure[N,2]

Figure of rotation curves for all given galaxies (N galaxies in total) and for the DC14 and NFW models.

5.11.3.34 `rotcurves_Einasto_check()`

```
def results.plots.rotcurves_Einasto_check (
    self,
    galaxies,
    fit_dict_in = fitting_dict_in,
    size = (20,20) )
```

Define the plot of the rotation curves for given galaxies and for the Einasto and NFW models.

This defines the plot of the rotation curves for all given galaxies and for the Einasto and NFW models. For comparison with Nicolas Loizeau and Glennys R. Farrar 2021 ApJL 920 L10 : <https://doi.org/10.3847/2041-8213/ac1bb7>

Parameters

<i>self</i>	object pointer
<i>fit_dict↔ _in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.
<i>size</i>	(optional) tuple Size of figure.

Returns

matplotlib.pyplot.figure[N,2]

Figure of rotation curves for all given galaxies (N galaxies in total) and for the Einasto and NFW models.

5.11.3.35 `rotcurves_psi_all()`

```
def results.plots.rotcurves_psi_all (
    self,
```

```
galaxies,
fit_dict_in = fitting_dict_in,
size = (20,20),
save_file = None )
```

Define the plot of the rotation curves for given galaxies and for the single ULDM and double ULDM models.

This defines the plot of the rotation curves for all given galaxies and for the single ULDM and double ULDM models.

Parameters

<i>self</i>	object pointer
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.
<i>size</i>	(optional) tuple Size of figure.
<i>save_file</i>	(optional) str File path to save plot. Default is None, in which case file is not saved.

Returns

matplotlib.pyplot.figure[N,2]

Figure of rotation curves for all given galaxies (N galaxies in total) and for the single and double ULDM (will correspond to either summed or matched model depending on value for [constants.fitting_dict.sol_match](#)).

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.12 results.results_CDM_all Class Reference

The class to obtain fit results for all CDM halos analyzed.

Public Member Functions

- def [__init__](#) (self, [model](#), [ULDM_fits](#)=False, [fit_dict_in](#)=[fitting_dict_in](#))
Define the constructor of the [results_CDM_all](#) class.
- def [fit](#) (self)
Define the fit results.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the *constants.fitting_dict* class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of *constants.fitting_dict.args_opts*
- **matched**
bool
True if performing ULDM fits and ULDM matched models
- **ULDM_fits**
bool
True if performing ULDM fits

5.12.1 Detailed Description

The class to obtain fit results for all CDM halos analyzed.

This class can be used to obtain results for all CDM halos analyzed.

120 galaxies in the SPARC catalog are used for CDM only fits, 93 galaxies used for ULDM fits with Einasto halo.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 __init__()

```
def results.results_CDM_all.__init__ (
    self,
    model,
    ULDM_fits = False,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the *results_CDM_all* class.

This defines the constructor of the *results_CDM_all* class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert • DC14 • Einasto • NEW
	Generated by Doxygen
<i>ULDM_fits</i>	(optional) bool True if performing fits for the ULDM models.
<i>fit_dict_in</i>	(optional) <i>constants.fitting_dict</i> instance Instance of the <i>constants.fitting_dict</i> class.

5.12.3 Member Function Documentation

5.12.3.1 fit()

```
def results.results_CDM_all.fit (
    self )
```

Define the fit results.

This defines the fit results for the assumed model for 120 galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 - 'Name' : ndarray[120],
 - 'fit' : ndarray[120],
 - 'Chi_sq' : ndarray[120],
 - 'BIC' : ndarray[120],
 - 'Mvir' : ndarray[120] (numpy.ndarray),
 - 'params' : dictionary,
- fit['Vbulge'] = same as fit['Vbulge_none'] Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
 - 'Name' = name of galaxy
 - 'fit' = [model_fit.model_fit.fit](#) object
 - 'Chi_sq' = reduced chi-squared
 - 'BIC' = Bayesian information criterion
 - 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
 - 'params' = {
 - 'c200' : ndarray[120] (numpy.ndarray) = halo concentration,
 - 'v200' : ndarray[120] (numpy.ndarray) = halo virial velocity,
 - 'MLd' : ndarray[120] (numpy.ndarray) = mass-to-light ratio of disk,
 - 'MLb' : ndarray[120] (numpy.ndarray) = mass-to-light ratio of bulge,
 - 'alpha' : ndarray[120] (numpy.ndarray) = Einasto halo parameter,
 - 'mstar' : ndarray[120] (numpy.ndarray) = mass of stellar component,
 - 'Vf' : ndarray[120] = maximum circular velocity,
 - 'mgas' : ndarray[120] = mass of gas component,

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.13 results.results_CDM_check Class Reference

The class to obtain fit results to check all CDM model implementation.

Public Member Functions

- `def __init__ (self, model, fit_dict_in=fitting_dict_in)`
Define the constructor of the `results_CDM_check` class.
- `def fit (self)`
Define the fit results.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the `constants.fitting_dict` class.

5.13.1 Detailed Description

The class to obtain fit results to check all CDM model implementation.

The class to obtain fit results for comparison with Pengfei Li et al 2020 ApJS 247 31 : <https://doi.org/10.3847/1538-4365/ab700e>

5.13.2 Constructor & Destructor Documentation

5.13.2.1 __init__()

```
def results.results_CDM_check.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the `results_CDM_check` class.

This defines the constructor of the `results_CDM_check` class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Burkert
	<ul style="list-style-type: none"> • DC14 • Einasto • NEW

5.13.3 Member Function Documentation

5.13.3.1 fit()

```
def results.results_CDM_check.fit (
    self )
```

Define the fit results.

This defines the fit results for the assumed model for all galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 - 'Name' : str[175],
 - 'Chi_sq' : ndarray[175],
 - 'params' : ndarray[175, 3 or 4],
 - 'Mvir' : float,
 - 'fit' : [model_fit.model_fit.fit](#) object,
- fit['Vbulge'] = same as 'Vbulge_none' with the change :
 - 'params' : ndarray[175, 4 or 5]
 Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
 - 'Name' = name of galaxy
 - 'Chi_sq' = reduced chi-squared
 - 'params' = fit parameters
 - for Burkert, DC14, and NFW without bulge component: [c200, v200, MLd]
 - for Burkert, DC14, and NFW with bulge component: [c200, v200, MLd, MLb]
 - for Einasto without bulge component: [c200, v200, MLd, alpha]
 - for Einasto with bulge component: [c200, v200, MLd, MLb, alpha]
 - 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
 - 'fit' = [model_fit.model_fit.fit](#) object

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.14 results.results_DC14_check Class Reference

The class to obtain fit results to check the DC14 and NFW model implementation.

Public Member Functions

- `def __init__ (self, model, fit_dict_in=fitting_dict_in)`
Define the constructor of the `results_DC14_check` class.
- `def fit (self)`
Define the fit results.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the `constants.fitting_dict` class.

5.14.1 Detailed Description

The class to obtain fit results to check the DC14 and NFW model implementation.

The class to obtain fit results for comparison with Monthly Notices of the Royal Astronomical Society, Volume 466, Issue 2, April 2017, Pages 1648–1668 : <https://doi.org/10.1093/mnras/stw3101>

5.14.2 Constructor & Destructor Documentation

5.14.2.1 __init__()

```
def results.results_DC14_check.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the `results_DC14_check` class.

This defines the constructor of the `results_DC14_check` class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • DC14
	<ul style="list-style-type: none"> • NFW
<i>fit_dict_in</i>	(optional) <code>constants.fitting_dict</code> instance Instance of the <code>constants.fitting_dict</code> class. Contains all the necessary rules and values to be used

5.14.3 Member Function Documentation

5.14.3.1 fit()

```
def results.results_DC14_check.fit (
    self )
```

Define the fit results.

This defines the fit results for the assumed model for 120 galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 - 'Name' : str[149],
 - 'Chi_sq' : ndarray[149],
 - 'params' : ndarray[149, 3],
 - 'Mvir' : float,
 - 'BIC' : float,
 - 'fit' : [model_fit.model_fit.fit](#) object,
- fit['Vbulge'] = same as 'Vbulge_none' with the change :
 - 'params' : ndarray[149, 4]
 Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
 - 'Name' = name of galaxy
 - 'Chi_sq' = reduced chi-squared
 - 'params' = fit parameters
 - for DC14, and NFW without/with bulge component: [c200, v200, MLd]
 - 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
 - 'BIC' = Bayesian information criterion (using lmfit definition)
 - 'fit' = [model_fit.model_fit.fit](#) object

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.15 results.results_Einasto_check Class Reference

The class to obtain fit results to check the Einasto and NFW model implementation.

Public Member Functions

- `def __init__ (self, model, fit_dict_in=fitting_dict_in)`
Define the constructor of the `results_Einasto_check` class.
- `def fit (self)`
Define the fit results.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the `constants.fitting_dict` class.

5.15.1 Detailed Description

The class to obtain fit results to check the Einasto and NFW model implementation.

The class to obtain fit results for comparison with Nicolas Loizeau and Glennys R. Farrar 2021 ApJL 920 L10 :
<https://doi.org/10.3847/2041-8213/ac1bb7>

5.15.2 Constructor & Destructor Documentation

5.15.2.1 __init__()

```
def results.results_Einasto_check.__init__ (
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the `results_Einasto_check` class.

This defines the constructor of the `results_Einasto_check` class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • Einasto • NFW
<i>fit_dict_in</i>	(optional) <code>constants.fitting_dict</code> instance Instance of the <code>constants.fitting_dict</code> class. Contains all the necessary rules and values to be used in fitting.

5.15.3 Member Function Documentation

5.15.3.1 fit()

```
def results.results_Einasto_check.fit (
    self )
```

Define the fit results.

This defines the fit results for the assumed model for 120 galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 - 'Name' : str[121],
 - 'Chi_sq' : ndarray[121],
 - 'params' : ndarray[121, 3 or 4],
 - 'Mvir' : float,
 - 'fit' : [model_fit.model_fit.fit](#) object,
- fit['Vbulge'] = same as 'Vbulge_none' with the change :
 - 'params' : ndarray[121, 4 or 5]
 Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
 - 'Name' = name of galaxy
 - 'Chi_sq' = reduced chi-squared
 - 'params' = fit parameters
 - for NFW without bulge component: [c200, v200, MLd]
 - for NFW with bulge component: [c200, v200, MLd, MLb]
 - for Einasto without bulge component: [c200, v200, MLd, alpha]
 - for Einasto with bulge component: [c200, v200, MLd, MLb, alpha]
 - 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
 - 'fit' = [model_fit.model_fit.fit](#) object

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.16 results.results_psi_multi_all Class Reference

The class to obtain fit results for all double flavored ULDM models analyzed.

Public Member Functions

- `def __init__ (self, fit_dict_in=fitting_dict_in)`
Define the constructor of the [results_psi_multi_all](#) class.
- `def fit (self)`
Define the fit results.

Public Attributes

- `fit_dict_in`
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class.
- `args_opts`
dictionary
Dictionary of rules to be assumed during fitting Instance of [constants.fitting_dict.args_opts](#)
- `mfree`
bool
Equal to true/false if soliton particle mass free/fixed in fitting procedure
- `cdmhalo`
str
Can be equal to :
- `matched`
bool
True if using ULDM matched models

5.16.1 Detailed Description

The class to obtain fit results for all double flavored ULDM models analyzed.

This class can be used to obtain results for all double flavored ULDM models analyzed. 93 galaxies in the SPARC catalog are used.

5.16.2 Constructor & Destructor Documentation

5.16.2.1 __init__()

```
def results.results_psi_multi_all.__init__ (
    self,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the [results_psi_multi_all](#) class.

This defines the constructor of the [results_psi_multi_all](#) class.

Parameters

<i>self</i>	object pointer
<i>fit_dict</i> <i>_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.16.3 Member Function Documentation

5.16.3.1 fit()

```
def results.results_psi_multi_all.fit (
    self )
```

Define the fit results.

This defines the fit results for the double flavor ULDM models for 93 galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 'Name' : ndarray[93],
 'fit' : ndarray[93],
 'Chi_sq' : ndarray[93],
 'BIC' : ndarray[93],
 'Mhalo_1' : ndarray[93],
 'Mhalo_2' : ndarray[93],
 'Mvir' : ndarray[93],
 'params' : dictionary,
}
- fit['Vbulge'] = same as fit['Vbulge_none'] Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
 - 'Name' = name of galaxy
 - 'fit' = [model_fit.model_fit.fit](#) object
 - 'Chi_sq' = reduced chi-squared
 - 'BIC' = Bayesian information criterion
 - 'Mhalo_1' = mass of ULDM halo one
 - 'Mhalo_2' = mass of ULDM halo two

- 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
- 'params' = {
 - 'c200' : ndarray[93] (numpy.ndarray) = halo one concentration,
 - 'c200_2' : ndarray[93] (numpy.ndarray) = halo two concentration,
 - 'v200' : ndarray[93] (numpy.ndarray) = halo one virial velocity,
 - 'v200_2' : ndarray[93] (numpy.ndarray) = halo two virial velocity,
 - 'MLd' : ndarray[93] (numpy.ndarray) = mass-to-light ratio of disk,
 - 'MLb' : ndarray[93] (numpy.ndarray) = mass-to-light ratio of bulge,
 - 'alpha' : ndarray[93] (numpy.ndarray) = Einasto halo parameter,
 - 'm22' : ndarray[93] (numpy.ndarray) = ULDM particle one mass,
 - 'm22_2' : ndarray[93] (numpy.ndarray) = ULDM particle two mass,
 - 'Msol' : ndarray[93] (numpy.ndarray) = soliton one mass,
 - 'Msol_2' : ndarray[93] (numpy.ndarray) = soliton two mass,
 - 'mstar' : ndarray[93] (numpy.ndarray) = mass of stellar component,
 - 'Vf' : ndarray[93] = maximum circular velocity,
 - 'mgas' : ndarray[93] = mass of gas component,

5.16.4 Member Data Documentation

5.16.4.1 cdmhalo

`results.results_psi_multi_all.cdmhalo`

str

Can be equal to :

- 'Burkert'
- 'DC14'
- 'Einasto'
- 'NFW'

The documentation for this class was generated from the following file:

- `pyfiles/fitting/results.py`

5.17 results.results_psi_single_all Class Reference

The class to obtain fit results for all single flavored ULDM models analyzed.

Public Member Functions

- `def __init__ (self, fit_dict_in=fitting_dict_in)`
Define the constructor of the `results_psi_single_all` class.
- `def fit (self)`
Define the fit results.

Public Attributes

- **fit_dict_in**
constants.fitting_dict instance
 Instance of the *constants.fitting_dict* class.
- **args_opts**
dictionary
 Dictionary of rules to be assumed during fitting Instance of *constants.fitting_dict.args_opts*
- **mfree**
bool
 Equal to true/false if soliton particle mass free/fixed in fitting procedure
- **cdmhalo**
str
 Can be equal to :
- **matched**
bool
 True if using ULDM matched models

5.17.1 Detailed Description

The class to obtain fit results for all single flavored ULDM models analyzed.

This class can be used to obtain results for all single flavored ULDM models analyzed.
 93 galaxies in the SPARC catalog are used.

5.17.2 Constructor & Destructor Documentation

5.17.2.1 __init__()

```
def results.results_psi_single_all.__init__ (
    self,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the *results_psi_single_all* class.

This defines the constructor of the *results_psi_single_all* class.

Parameters

<i>self</i>	object pointer
<i>fit_dict_in</i>	(optional) <i>constants.fitting_dict</i> instance Instance of the <i>constants.fitting_dict</i> class. Contains all the necessary rules and values to be used in fitting.

5.17.3 Member Function Documentation

5.17.3.1 fit()

```
def results.results_psi_single_all.fit (
    self )
```

Define the fit results.

This defines the fit results for the single flavor ULDM models for 93 galaxies in the SPARC catalog.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

dictionary

Dictionary of resulting fit parameters. Results are as follows : fit = {'Vbulge_none' : {...}, 'Vbulge' : {...}} where 'Vbulge_none'/'Vbulge' corresponds to galaxies without/with a bulge component.

- fit['Vbulge_none'] = {
 'Name' : ndarray[93],
 'fit' : ndarray[93],
 'Chi_sq' : ndarray[93],
 'BIC' : ndarray[93],
 'Mhalo' : ndarray[93],
 'Mvir' : ndarray[93],
 'params' : dictionary,
 }
- fit['Vbulge'] = same as fit['Vbulge_none'] Each of the components of fit['Vbulge_none'] and fit['Vbulge'] are :
- 'Name' = name of galaxy
- 'fit' = [model_fit.model_fit.fit](#) object
- 'Chi_sq' = reduced chi-squared
- 'BIC' = Bayesian information criterion
- 'Mhalo' = mass of ULDM halo
- 'Mvir' = M200 (mass enclosed within radius R200) in units of solar masses
- 'params' = {
 'c200' : ndarray[93] (numpy.ndarray) = halo concentration,
 'v200' : ndarray[93] (numpy.ndarray) = halo virial velocity,
 'MLd' : ndarray[93] (numpy.ndarray) = mass-to-light ratio of disk,
 'MLb' : ndarray[93] (numpy.ndarray) = mass-to-light ratio of bulge,
 'alpha' : ndarray[93] (numpy.ndarray) = Einasto halo parameter,
 'm22' : ndarray[93] (numpy.ndarray) = ULDM particle mass,
 'Msol' : ndarray[93] (numpy.ndarray) = soliton mass,
 'mstar' : ndarray[93] (numpy.ndarray) = mass of stellar component,
 'Vf' : ndarray[93] = maximum circular velocity,
 'mgas' : ndarray[93] = mass of gas component,
 }

5.17.4 Member Data Documentation

5.17.4.1 cdmhalo

results.results_psi_single_all.cdmhalo

str

Can be equal to :

- 'Burkert'
- 'DC14'
- 'Einasto'
- 'NFW'

The documentation for this class was generated from the following file:

- pyfiles/fitting/results.py

5.18 alp_funcs.soliton Class Reference

The class containing mass functions for the ULDM halo models.

Public Member Functions

- def `__init__` (self, model, fit_dict_in=fitting_dict_in)
Define the constructor of the soliton class.
- def `mass` (self, params, r)
Define the mass profile of the ULDM galactic structure.
- def `Mhalo` (self, params)
Define the total mass of the outer halo.
- def `Mvir` (self, params)
Define the total galactic DM halo mass.

Public Attributes

- **model**
str
Model to assume for DM halo.
- **fit_dict_in**
constants.fitting_dict instance
Instance of the *constants.fitting_dict* class.
- **args_opts**
dictionary
Dictionary of rules to be assumed during fitting Instance of *constants.fitting_dict.args_opts*
- **matched**
bool
Denotes how to combine the soliton and outer halo.
- **mfree**
float
Denotes how to treat soliton particle mass in fitting procedure.
- **m22**
float
Soliton particle mass in units of 10^{-22} eV.
- **m22_2**
float
Soliton particle mass two in units of 10^{-22} eV.
- **cdmhalo**
float
Denotes which CDM profile to use for outer halo in ULDM galactic structure.
- **base_funcs_in**
alp_funcs.base_funcs instance
Instance of the *alp_funcs.base_funcs* class
- **halo_init**
cdm_funcs.halo instance
Instance of the *cdm_funcs.halo* class

5.18.1 Detailed Description

The class containing mass functions for the ULDM halo models.

5.18.2 Constructor & Destructor Documentation

5.18.2.1 `__init__()`

```
def alp_funcs.soliton.__init__(
    self,
    model,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the soliton class.

This defines the constructor of the soliton class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none">• psi_single• psi_multi
<i>fit_dict</i> <i>_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.18.3 Member Function Documentation

5.18.3.1 mass()

```
def alp_funcs.soliton.mass (
    self,
    params,
    r )
```

Define the mass profile of the ULDM galactic structure.

This defines the mass of the ULDM galactic structure (soliton and outer halo) at a given radius given the model parameters.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .
<i>r</i>	ndarray[N] Numpy array of N radii values in units of kpc.

Returns

ndarray[N]
Numpy array of N mass values in units of solar mass.

5.18.3.2 Mhalo()

```
def alp_funcs.soliton.Mhalo (
    self,
    params )
```

Define the total mass of the outer halo.

This defines the total mass of the outer halo assuming the model parameters.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using alp_params.soliton.params .

Returns

float

Total mass of the outer halo in units of solar mass.

5.18.3.3 Mvir()

```
def alp_funcs.soliton.Mvir (
    self,
    params )
```

Define the total galactic DM halo mass.

This defines the total mass of the galactic DM given the model parameters.

Parameters

<i>self</i>	object pointer
<i>params</i>	Imfit.Parameters instance Instance of the Imfit.Parameters class. All model parameters contained here. This can be created using cdm_params.halo.params .

Returns

float

Total mass of the galactic DM in units of solar mass.

5.18.4 Member Data Documentation

5.18.4.1 cdmhalo

```
alp_funcs.soliton.cdmhalo
```

float

Denotes which CDM profile to use for outer halo in ULDM galactic structure.

Equal to constants.fitting_dict.sol_cdmhalo

5.18.4.2 matched

`alp_funcs.soliton.matched`

bool

Denotes how to combine the soliton and outer halo.

Equal [constants.fitting_dict.sol_match](#)

5.18.4.3 mfree

`alp_funcs.soliton.mfree`

float

Denotes how to treat soliton particle mass in fitting procedure.

Equal to [constants.fitting_dict.sol_mfree](#)

The documentation for this class was generated from the following file:

- `pyfiles/models/alps/alp_funcs.py`

5.19 alp_params.soliton Class Reference

The class containing all parameters for fitting the ULDM halo models.

Public Member Functions

- `def __init__ (self, model, data, fit_dict_in=fitting_dict_in)`
Define the constructor of the soliton class.
- `def params (self)`
Define the parameters to be assumed.

Public Attributes

- **`model`**
str
Model to assume for DM halo.
- **`data`**
dictionary
Dictionary containing all the necessary data for a given galaxy.
- **`fit_dict_in`**
[constants.fitting_dict](#) instance
Instance of the [constants.fitting_dict](#) class.
- **`args_opts`**
dictionary
Dictionary of rules to be assumed during fitting Instance of [constants.fitting_dict.args_opts](#)
- **`params_vals`**

- dictionary*
Dictionary of variable values to be used during fitting.
- **c200**
 - list*
Values for c200 in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **v200**
 - list*
Values for V200 in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **v200fac**
 - list*
Values for v200fac in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **MLD**
 - list*
Values for MLD in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **MLB**
 - list*
Values for MLB in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **alpha**
 - list*
Values for alpha in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **msol**
 - float*
Values for msol in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **m22**
 - float*
Values for m22 in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **msol_2**
 - float*
Values for msol two in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **m22_2**
 - float*
Values for m22 two in the form [starting value, min, max] Contained in [constants.fitting_dict.params_vals](#)
- **matched**
 - bool*
Denotes how to combine the soliton and outer halo.
- **mfree**
 - float*
Denotes how to treat soliton particle mass in fitting procedure.
- **cdmhalo**
 - float*
Denotes which CDM profile to use for outer halo in ULDM galactic structure.

5.19.1 Detailed Description

The class containing all parameters for fitting the ULDM halo models.

This class contains all parameters necessary to perform the fitting procedures for all ULDM halo models.

5.19.2 Constructor & Destructor Documentation

5.19.2.1 `__init__()`

```
def alp_params.soliton.__init__ (
    self,
    model,
    data,
    fit_dict_in = fitting_dict_in )
```

Define the constructor of the soliton class.

This defines the constructor of the soliton class.

Parameters

<i>self</i>	object pointer
<i>model</i>	str Model to assume for DM halo. Can be equal to : <ul style="list-style-type: none"> • <code>psi_single</code> • <code>psi_multi</code>
<i>data</i>	dictionary Dictionary containing all the necessary data for a given galaxy. This can be created using galaxy.galaxy .
<i>fit_dict_in</i>	(optional) constants.fitting_dict instance Instance of the constants.fitting_dict class. Contains all the necessary rules and values to be used in fitting.

5.19.3 Member Function Documentation

5.19.3.1 `params()`

```
def alp_params.soliton.params (
    self )
```

Define the parameters to be assumed.

This defines the parameters to be assumed during the fitting procedures for all ULDM halos.

Parameters

<i>self</i>	object pointer
-------------	----------------

Returns

Imfit.Parameters instance Instance of the Imfit.Parameters class

5.19.4 Member Data Documentation

5.19.4.1 cdmhalo

`alp_params.soliton.cdmhalo`

float

Denotes which CDM profile to use for outer halo in ULDM galactic structure.

Equal to `constants.fitting_dict.sol_cdmhalo`

5.19.4.2 data

`alp_params.soliton.data`

dictionary

Dictionary containing all the necessary data for a given galaxy.

This can be created using [galaxy.galaxy](#).

5.19.4.3 matched

`alp_params.soliton.matched`

bool

Denotes how to combine the soliton and outer halo.

Equal [constants.fitting_dict.sol_match](#)

5.19.4.4 mfree

`alp_params.soliton.mfree`

float

Denotes how to treat soliton particle mass in fitting procedure.

Equal to [constants.fitting_dict.sol_mfree](#)

5.19.4.5 params_vals

`alp_params.soliton.params_vals`

dictionary

Dictionary of variable values to be used during fitting.

Instance of [constants.fitting_dict.params_vals](#)

The documentation for this class was generated from the following file:

- `pyfiles/models/alps/alp_params.py`

Index

- `__init__`
 - `alp_funcs.base_funcs`, 20
 - `alp_funcs.soliton`, 102
 - `alp_params.soliton`, 106
 - `cdm_funcs.base_funcs`, 25
 - `cdm_funcs.halo`, 43
 - `cdm_params.base_funcs`, 28
 - `cdm_params.halo`, 47
 - `constants.fitting_dict`, 31
 - `galaxy.galaxy`, 36
 - `halo.halo`, 50
 - `model_fit.grar_fit`, 39
 - `model_fit.model_fit`, 53
 - `results.plots`, 59
 - `results.results_CDM_all`, 88
 - `results.results_CDM_check`, 90
 - `results.results_DC14_check`, 92
 - `results.results_Einasto_check`, 94
 - `results.results_psi_multi_all`, 96
 - `results.results_psi_single_all`, 99
- `abund_match_rel`
 - `model_fit`, 14
- `alp_funcs`, 7
- `alp_funcs.base_funcs`, 19
 - `__init__`, 20
 - `cdmhalo`, 23
 - `mass_sol_init`, 20
 - `matched`, 23
 - `mfree`, 24
 - `msol`, 21
 - `rc`, 21
 - `rhoc`, 22
 - `xc`, 23
- `alp_funcs.soliton`, 101
 - `__init__`, 102
 - `cdmhalo`, 104
 - `mass`, 103
 - `matched`, 104
 - `mfree`, 105
 - `Mhalo`, 103
 - `Mvir`, 104
- `alp_params`, 8
 - `alphamatched`, 9
- `alp_params.soliton`, 105
 - `__init__`, 106
 - `cdmhalo`, 108
 - `data`, 108
 - `matched`, 108
 - `mfree`, 108
 - `params`, 107
 - `params_vals`, 108
- `alphamatched`
 - `alp_params`, 9
- `args_opts`
 - `constants.fitting_dict`, 33
 - `model_fit.model_fit`, 57
- `bic`
 - `model_fit.model_fit`, 54
- `BIC_CDM`
 - `results.plots`, 60
- `BIC_diffs_CDM`
 - `results.plots`, 61
- `BIC_psi_mfix_ex`
 - `results.plots`, 61
- `BIC_psi_mfree`
 - `results.plots`, 63
- `BTFR`
 - `model_fit`, 16
- `cdm_funcs`, 9
- `cdm_funcs.base_funcs`, 24
 - `__init__`, 25
 - `mass_frac_dc14`, 25
 - `mass_num`, 28
 - `rc`, 26
 - `rhoc`, 26
 - `xc`, 27
- `cdm_funcs.halo`, 43
 - `__init__`, 43
 - `mass`, 44
 - `mass_num`, 46
 - `Mvir`, 45
- `cdm_params`, 10
- `cdm_params.base_funcs`, 28
 - `__init__`, 28
 - `v200min_dc14`, 29
 - `v200min_frac_dc14check`, 29
- `cdm_params.halo`, 46
 - `__init__`, 47
 - `data`, 48
 - `params`, 48
 - `params_vals`, 48
- `cdmhalo`
 - `alp_funcs.base_funcs`, 23
 - `alp_funcs.soliton`, 104
 - `alp_params.soliton`, 108
 - `results.results_psi_multi_all`, 98
 - `results.results_psi_single_all`, 101

- chi_box_Einasto_checks
 - results.plots, 63
- chi_CDM
 - results.plots, 64
- chi_dist_CDM
 - results.plots, 65
- chi_dist_CDM_checks
 - results.plots, 65
- chi_dist_DC14_checks
 - results.plots, 66
- chi_dist_Einasto_checks
 - results.plots, 67
- chi_dist_psi_mfix_ex
 - results.plots, 67
- chi_dist_psi_mfree
 - results.plots, 68
- chi_gal_psi_mfix
 - results.plots, 69
- chi_psi_mfix
 - results.plots, 70
- chi_psi_mfix_ex
 - results.plots, 71
- chi_psi_mfree
 - results.plots, 72
- conc_mass_rel_Du
 - model_fit, 16
- conc_mass_rel_Wa
 - model_fit, 17
- constants, 11
 - standard, 12
- constants.fitting_dict, 30
 - __init__, 31
 - args_opts, 33
 - params_vals, 34
- data
 - alp_params.soliton, 108
 - cdm_params.halo, 48
 - galaxy.galaxy, 37
 - halo.halo, 52
- df
 - galaxy.galaxy, 37
- df1
 - galaxy.galaxy, 37
- fit
 - model_fit.grar_fit, 39
 - model_fit.model_fit, 54
 - results.results_CDM_all, 89
 - results.results_CDM_check, 91
 - results.results_DC14_check, 93
 - results.results_Einasto_check, 95
 - results.results_psi_multi_all, 97
 - results.results_psi_single_all, 100
- g_bar
 - model_fit.grar_fit, 40
- g_rar
 - model_fit.grar_fit, 41
- g_tot
 - model_fit.grar_fit, 41
- galaxy, 12
 - galaxy.galaxy, 36
 - __init__, 36
 - data, 37
 - df, 37
 - df1, 37
- grar_model
 - model_fit.grar_fit, 42
- halo, 13
 - halo.halo, 49
 - __init__, 50
 - data, 52
 - halo_init, 52
 - mass, 50
 - params, 52
 - params_vals, 52
 - velocity, 51
- halo_init
 - halo.halo, 52
- mass
 - alp_funcs.soliton, 103
 - cdm_funcs.halo, 44
 - halo.halo, 50
- mass_frac_dc14
 - cdm_funcs.base_funcs, 25
- mass_num
 - cdm_funcs.base_funcs, 28
 - cdm_funcs.halo, 46
- mass_sol_init
 - alp_funcs.base_funcs, 20
- matched
 - alp_funcs.base_funcs, 23
 - alp_funcs.soliton, 104
 - alp_params.soliton, 108
- mfree
 - alp_funcs.base_funcs, 24
 - alp_funcs.soliton, 105
 - alp_params.soliton, 108
- Mhalo
 - alp_funcs.soliton, 103
- MLd_degen_CDM
 - results.plots, 73
- model_fit, 13
 - abund_match_rel, 14
 - BTFR, 16
 - conc_mass_rel_Du, 16
 - conc_mass_rel_Wa, 17
- model_fit.grar_fit, 38
 - __init__, 39
 - fit, 39
 - g_bar, 40
 - g_rar, 41
 - g_tot, 41
 - grar_model, 42
- model_fit.model_fit, 53

- `__init__`, 53
 - `args_opts`, 57
 - `bic`, 54
 - `fit`, 54
 - `plot`, 55
 - `residual`, 55
 - `velocity_tot`, 56
- `msol`
 - `alp_funcs.base_funcs`, 21
- `Msol_psi_mfix`
 - `results.plots`, 74
- `Msol_psi_mfix_ex`
 - `results.plots`, 74
- `Msol_psi_mfree`
 - `results.plots`, 75
- `Mvir`
 - `alp_funcs.soliton`, 104
 - `cdm_funcs.halo`, 45
- `params`
 - `alp_params.soliton`, 107
 - `cdm_params.halo`, 48
 - `halo.halo`, 52
- `params_dist_CDM`
 - `results.plots`, 77
- `params_dist_CDM_checks`
 - `results.plots`, 78
- `params_dist_DC14_checks`
 - `results.plots`, 78
- `params_dist_Einasto_checks`
 - `results.plots`, 79
- `params_dist_psi_mfix_ex`
 - `results.plots`, 79
- `params_dist_psi_mfree`
 - `results.plots`, 80
- `params_scatter_Einasto_checks`
 - `results.plots`, 81
- `params_vals`
 - `alp_params.soliton`, 108
 - `cdm_params.halo`, 48
 - `constants.fitting_dict`, 34
 - `halo.halo`, 52
- `plot`
 - `model_fit.model_fit`, 55
- `rc`
 - `alp_funcs.base_funcs`, 21
 - `cdm_funcs.base_funcs`, 26
- `relations_CDM`
 - `results.plots`, 82
- `relations_psi_mfix_ex`
 - `results.plots`, 82
- `relations_psi_mfree`
 - `results.plots`, 83
- `residual`
 - `model_fit.model_fit`, 55
- `results`, 17
- `results.plots`, 57
- `__init__`, 59
- `BIC_CDM`, 60
- `BIC_diffs_CDM`, 61
- `BIC_psi_mfix_ex`, 61
- `BIC_psi_mfree`, 63
- `chi_box_Einasto_checks`, 63
- `chi_CDM`, 64
- `chi_dist_CDM`, 65
- `chi_dist_CDM_checks`, 65
- `chi_dist_DC14_checks`, 66
- `chi_dist_Einasto_checks`, 67
- `chi_dist_psi_mfix_ex`, 67
- `chi_dist_psi_mfree`, 68
- `chi_gal_psi_mfix`, 69
- `chi_psi_mfix`, 70
- `chi_psi_mfix_ex`, 71
- `chi_psi_mfree`, 72
- `MLd_degen_CDM`, 73
- `Msol_psi_mfix`, 74
- `Msol_psi_mfix_ex`, 74
- `Msol_psi_mfree`, 75
- `params_dist_CDM`, 77
- `params_dist_CDM_checks`, 78
- `params_dist_DC14_checks`, 78
- `params_dist_Einasto_checks`, 79
- `params_dist_psi_mfix_ex`, 79
- `params_dist_psi_mfree`, 80
- `params_scatter_Einasto_checks`, 81
- `relations_CDM`, 82
- `relations_psi_mfix_ex`, 82
- `relations_psi_mfree`, 83
- `rotcurves_CDM_all`, 84
- `rotcurves_CDM_check`, 85
- `rotcurves_DC14_check`, 85
- `rotcurves_Einasto_check`, 86
- `rotcurves_psi_all`, 86
- `results.results_CDM_all`, 87
- `__init__`, 88
- `fit`, 89
- `results.results_CDM_check`, 90
- `__init__`, 90
- `fit`, 91
- `results.results_DC14_check`, 92
- `__init__`, 92
- `fit`, 93
- `results.results_Einasto_check`, 93
- `__init__`, 94
- `fit`, 95
- `results.results_psi_multi_all`, 96
- `__init__`, 96
- `cdmhalo`, 98
- `fit`, 97
- `results.results_psi_single_all`, 98
- `__init__`, 99
- `cdmhalo`, 101
- `fit`, 100
- `rhoc`
 - `alp_funcs.base_funcs`, 22
 - `cdm_funcs.base_funcs`, 26

- rotcurves_CDM_all
 - results.plots, [84](#)
- rotcurves_CDM_check
 - results.plots, [85](#)
- rotcurves_DC14_check
 - results.plots, [85](#)
- rotcurves_Einasto_check
 - results.plots, [86](#)
- rotcurves_psi_all
 - results.plots, [86](#)
- standard
 - constants, [12](#)
- v200min_dc14
 - cdm_params.base_funcs, [29](#)
- v200min_frac_dc14check
 - cdm_params.base_funcs, [29](#)
- velocity
 - halo.halo, [51](#)
- velocity_tot
 - model_fit.model_fit, [56](#)
- xc
 - alp_funcs.base_funcs, [23](#)
 - cdm_funcs.base_funcs, [27](#)