
The Exoplanet Simple Orbit Fitting Toolbox "model" Documentation

Release 1.0

Kyle Mede, Timothy D. Brandt

December 19, 2016

| | | |
|----------|---|-----------|
| 1 | Examples to show how to use ExoSOFtmodel | 3 |
| 2 | The primary objects/classes and likelihood function for ExoSOFtmodel | 5 |
| 3 | Cython tools for ExoSOFtmodel | 7 |
| 4 | Priors for ExoSOFtmodel | 9 |
| 5 | Simple start-up tools for ExoSOFtmodel | 11 |
| 6 | Indices and tables | 13 |
| | Python Module Index | 15 |

Contents:

Examples to show how to use ExoSOFTmodel

Contents:

The primary objects/classes and likelihood function for ExoSFTmodel

Contents:

class ExoSFTmodel.model.**ExoSFTdata** (*epochs_di, epochs_rv, rapa, rapa_err, decsa, decsa_err, rv, rv_err, rv_inst_num, data_mode, pasa=False*)

Bases: object

An object to contain all the necessary data arrays and parameters to calculate matching predicted data with the model. All member variables will remain constant throughout.

Notes: -Except for `rv_inst_num` array, all other arrays must be ndarrays of double precision floating point numbers (`dtype=np.dtype('d')`).

-Arrays, `epochs_di`, `rapa`, `rapa_err`, `decsa`, and `decsa_err` must all have same length. -Arrays, `epochs_rv`, `rv`, `rv_err` and `rv_inst_num` must all have same length.

Inputs: `rv_inst_num` = ndarray of positive signed or unsigned integers, of same length as `epochs_rv`, `rv`, and `rv_err`.

class ExoSFTmodel.model.**ExoSFTmodel** (*sd*)

Bases: object

class ExoSFTmodel.model.**ExoSFTparams** (*omega_offset_di, omega_offset_rv, vary_tc, tc_equal_to, di_only, low_ecc, range_maxs, range_mins, num_offsets*)

Bases: object

| | | | | |
|---|--------------------------|----------------------------|--------------------------|-------|
| | Directly Varried | Model Inputs | Stored Parameters | |
| | <code>direct_pars</code> | <code>model_in_pars</code> | <code>stored_pars</code> | |
| i | Parameter | Parameter | Parameter | units |
| 0 | Mass of Primary (m1) | m1 | m1 | Msun |

.\$\$ FILL THIS OUT!!!!

check_range ()

Determine if all parameters in the full list are within their allowed ranges. Range arrays correspond to parameters in: [m1, m2, parallax, long_an, e, to/tc, p, inc, arg_peri, v1,v2,...]

make_model_in ()

Convert directly varied parameters into a comprehensive list of those used as inputs to during model calculations.

model_in_params: [m1,m2,parallax,long_an,e,to,tc,p,inc,arg_peri,arg_peri_di,arg_peri_rv,a_tot_au,K]

make_stored (*chi_squared*)

Push values in model_in_params, offsets and the resulting chi_squared_3d into an array to be stored on disk during ExoSOFT. Not sure how to make this work with emcee or other tools...

stored_to_direct (*pars*)

take a set of parameters matching 'stored_pars' and make the directly varied versions matching 'direct_pars'. Note: direct_pars: [m1,m2,parallax,long_an,e OR sqrt(e)*sin(arg_peri),to/tc,p,inc,arg_peri OR sqrt(e)*cos(arg_peri),v1,v2...] stored_pars: [m1,m2,parallax,long_an,e,to,tc,p,inc,arg_peri,a_tot_au,chi_sqr,K,v1,v2...]

ExoSOFTmodel.model.ln_posterior (*pars, Model*)

Calculates the likelihood for a given set of inputs. Then calculate the natural logarithm of the posterior probability.

-Model is of type ExoSOFTmodel. Currently just holds resulting fit values. -Data is of type ExoSOFTdata, containing all input data and params to produce predicted values of matching units, and arrays for predicted values. -Params is of type ExoSOFTparams, an class containing functions for calculating versions of the 'pars' used as model inputs, and a version that would be for storing to disk when ran in ExoSOFT. -Priors is of type ExoSOFTpriors, containing funtions for each parameter's prior, a function calculate combined prior given list of params, and any variables necessary for those calculations.

Cython tools for ExoSOFTmodel

Contents:

Priors for ExoSOFTmodel

Contents:

```
class ExoSOFTmodel.priors.ExoSOFTpriors (ecc_prior=True, p_prior=True, inc_prior=True,  
                                         m1_prior=True, m2_prior=True, para_prior=True,  
                                         inc_min=0.0, inc_max=180.0, p_min=0.0,  
                                         p_max=300.0, para_est=0, para_err=0, m1_est=0,  
                                         m1_err=0, m2_est=0, m2_err=0)
```

Bases: object

Choices: inclination: (None,True,'sin','cos') True indicates default of 'sin' eccentricity: (None,True) True indicates default of '2e' period: (None,True) True indicates default of 'log' mass of primary: (None,True,'IMF','PDMF') True indicates default of 'PDMF' mass of companion: (None,True,'IMF','PDMF','CMF') True indicates default of 'CMF' parallax: (None,True) True indicates default of 'gauss'

cmf_prior (*m2, m1*)

from equation 8 of Metchev & Hillenbrand 2009

ecc_prior_fn (*ecc*)

gaussian (*x, mu, sig*)

imf_prior (*m*)

From table 1 of Chabrier et al. 2003

inc_prior_fn (*inc*)

m1_prior_fn (*mass*)

m2_prior_fn (*m2, m1*)

p_prior_fn (*p*)

para_prior_fn (*para*)

pdmf_prior (*m*)

From table 1 of Chabrier et al. 2003

priors (*pars*)

Combined priors for a single step in the chain. This can be called to form the priors ratio OR when accounting for the priors when producing the posteriors during post-processing.

Input array needs the first elements to be: [m1, m2, parallax, long_an, e, to, tc, p, inc, arg_peri] This can be the full 'model_in_pars', 'stored_pars' or a truncated version with just these parameters.

priors_ratio (*pars_prop, pars_last*)

Calculates the priors ratio.

Input arrays need the first elements to be: [m1, m2, parallax, long_an, e, to, tc, p, inc, arg_peri] This can be the full 'model_in_pars' or a truncated version with just these parameters.

test_priors (*pars_prop*, *pars_last*)

Just for testing no errors occur while trying to calculate the priors ratio. Nothing returned.

Simple start-up tools for ExoSOFTmodel

Contents:

`ExoSOFTmodel.tools.load_di_data(filename)`

Load the astrometry data into a numpy array.

file format: title column headers data . . .

Data must be in the columns: obsDate[JD] x[""] x_error[""] y[""] y_error[""] OR if pasa key in settingsAdvanced ==True, then: obsDate[JD] PA[deg] PA_error[deg] SA[""] SA_error[""]

`ExoSOFTmodel.tools.load_rv_data(filename)`

Load the radial velocity data into a numpy array. Provided jitter values will be added in quadrature with the errors.

file format: title column headers data . . Empty line between data sets data . .

Data must be in the columns: obsDate[JD] RV[m/s] RV_error[m/s] jitter[m/s] datasetNumber[int] NOTE: datasetNumber is optional, if not provided they will be automatically set to 0,1,2... following the order of the data in the file.

If jitter is not provided, it will be assumed zero.

`ExoSOFTmodel.tools.load_settings_dict(settings_filename)`

A way to load in settings from a settings an ExoSOFT type settings file.

Users can choose to copy and modify the one provided in the 'examples' directory to match the system they are working on, or write their own function like this to load in the required parameters/arrays... that comprise all the inputs for ExoSOFTpriors, ExoSOFTdata and ExoSOFTparams.

NOTE: THE DICTIONARY KEY WORDS USED HERE DO NOT WORK FOR OLD EXOSOFT SETTINGS FILES. NEED TO CHANGE ALL OF THEM TO THIS FORMAT EVENTUALLY!!!!

`ExoSOFTmodel.tools.make_starting_params(pars, n, scale=0.01)`

Creates an array of starting guesses tightly centered on the input parameters.

Indices and tables

- `genindex`
- `modindex`
- `search`

e

`ExoSFTmodel.cytools`, [7](#)
`ExoSFTmodel.model`, [5](#)
`ExoSFTmodel.priors`, [9](#)
`ExoSFTmodel.tools`, [11](#)

C

`check_range()` (ExoSOFtmodel.model.ExoSOFtparams method), 5

`cmf_prior()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

E

`ecc_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`ExoSOFtdata` (class in ExoSOFtmodel.model), 5

`ExoSOFtmodel` (class in ExoSOFtmodel.model), 5

`ExoSOFtmodel.cytools` (module), 7

`ExoSOFtmodel.model` (module), 5

`ExoSOFtmodel.priors` (module), 9

`ExoSOFtmodel.tools` (module), 11

`ExoSOFtparams` (class in ExoSOFtmodel.model), 5

`ExoSOFtpriors` (class in ExoSOFtmodel.priors), 9

G

`gaussian()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

I

`imf_prior()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`inc_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

L

`ln_posterior()` (in module ExoSOFtmodel.model), 6

`load_di_data()` (in module ExoSOFtmodel.tools), 11

`load_rv_data()` (in module ExoSOFtmodel.tools), 11

`load_settings_dict()` (in module ExoSOFtmodel.tools), 11

M

`m1_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`m2_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`make_model_in()` (ExoSOFtmodel.model.ExoSOFtparams method), 5

`make_starting_params()` (in module ExoSOFtmodel.tools), 11

`make_stored()` (ExoSOFtmodel.model.ExoSOFtparams method), 5

P

`p_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`para_prior_fn()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`pdmf_prior()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`priors()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

`priors_ratio()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 9

S

`stored_to_direct()` (ExoSOFtmodel.model.ExoSOFtparams method), 6

T

`test_priors()` (ExoSOFtmodel.priors.ExoSOFtpriors method), 10