swprepost

Release 2.0.0

May 17, 2022

Contents

swprepost is a Python package for surface wave inversion pre- and post-processing. It includes 12 class definitions to handle all aspect of the surface wave inversion process.

This package and the classes therein are actively being developed, so if you do not see a feature you would like it may very well be under development and released in the near future. To be notified of future releases, you can either `watch` the repository on Github or `Subscribe to releases` on the Python Package Index (PyPI).

CHAPTER 1

Contents:

1.1 Installation

`pip install swprepost` or `pip install swprepost --upgrade`

`pip` will handle the rest!

1.2 API Reference

1.2.1 Curve

The Curve class definition.

class Curve(*x*, *y*, *check_fxn=None*)

Bases: `object`

Base class for handling sets of x, y coordinates.

Variables

- **_x** (*ndarray*) – 1D array of x coordinates defining the curve. These should, in general, not be accessed directly.
- **_y** (*ndarray*) – Same as *_x* but for the y coordinates of the curve.

__init__(*x*, *y*, *check_fxn=None*)

Initialize a curve object from x, y coordinates.

Parameters

- **x, y** (*iterable*) – Iterables of the same size defining the curve's x and y coordinates.
- **check_fxn** (*function, optional*) – Function that takes an x, y pair, and checks if they are valid.

If they are valid the function returns *None* otherwise raises a *ValueError*, default is *None* meaning no function is used to check the *x* and *y* values.

Returns *Curve* – Instantiated *Curve* object.

Raises

- `IndexError` – If *x* and *y* do not have the same length.
- `ValueError` – If *check_fxn* is defined and any *x*, *y* pair fails to meet the defined criteria.

classmethod check_input(*x*, *y*, *check_fxn=None*)
Check inputs comply with the required formatting.

static check_types(*x*, *y*)
Check type of x and y

Specifically:

1. Cast x and y to *ndarray* of type *double*.
2. Check x and y have the same length.

static check_values(*x*, *y*, *check_fxn*)
Apply custom checking function on the values of *x* and *y*.

Parameters

- **x, y** (*iterable*) – x and y value of curve respectively.
- **check_fxn** (*function*) – Function that takes an x, y pair, checks if they are valid. If they are valid the function returns *None* otherwise raises a *ValueError*.

Returns *None* – If *x* and *y* pass

Raises `ValueError` – If *x* and *y* fail.

resample(*xx*, *inplace=False*, *interp1d_kwargs=None*, *res_fxn=None*)
Resample Curve at select x values.

Parameters

- **xx** (*ndarray*) – 1D array containing the locations in terms of x, of the desired interpolated y values.
- **inplace** (*bool, optional*) – Indicates whether resampling should be done in-place. If inplace the, attributes *_x* and *_y* are overwritten. Otherwise the new values are returned, default is *False* resampling is not done inplace.
- **interp1d_settings** (*dict, optional*) – Settings for use with the *interp1d* function from *scipy*. See documentation here for details.
- **res_fxn** (*function, optional*) – Define a custom resampling function. It should accept an ndarray of resampling x-coordinates and return the interpolated y-coordinates as an ndarray.

Returns *None or (xx, yy)* – *None*, if *inplace=True*; *_x* and *_y* will be updated. *(xx, yy)* if *inplace=False*.

classmethod resample_function(*x*, *y*, ***kwargs*)
Wrapper for *interp1d* from *scipy*.

1.2.2 CurveUncertain

Definition of CurveUncertain.

class CurveUncertain (*x*, *y*, *yerr=None*, *xerr=None*)
Bases: *swprepost.curve.Curve*

Curve object with arbitrary uncertainty in terms of x and y.

Variables

- **_isyerr** (*_isxerr,*) – Flags to indicate if x and y error has been provided.
- **_yerr** (*_xerr,*) – Vector defining the error in x and y respectively.

__init__ (*x*, *y*, *yerr=None*, *xerr=None*)
Initialize a new *CurveUncertain* object.

Parameters

- **x, y** (*iterable*) – x and y coordinate which define the curves central trend.
- **yerr, xerr** (*iterable, optional*) – Relative error in the y- and x-direction respectively, default is *None* indicating no error is defined.

Returns *CurveUncertain* – Initialized *CurveUncertain* object.

Raises `IndexError` – If size of x, y, yerr (if provided) and xerr (if provided) are inconsistent.

resample (*xx*, *inplace=False*, *interp1d_kwargs=None*, *res_fxn=None*)
Resample curve and its associated uncertainty.

Parameters

- **xx** (*ndarray*) – Desired x values after resampling.
- **inplace** (*bool, optional*) – Indicates whether resampling is performed inplace and the objects attributes are updated or if calculated values are returned.
- **interp1d_settings** (*dict, optional*) – Settings for use with the *interp1d* function from *scipy*. See documentation here for details.
- **res_fxn** (*tuple of functions, optional*) – Functions to define the resampling of the central x and y values, xerr and yerr respectively, default is *None* indicating default resampling function is used.

Returns *None or Tuple* – If *inplace=True*, returns *None*, instead update attributes *_x*, *_y*, *_xerr*, and *_yerr* if they exist. If *inplace=False*, returns *Tuple* of the form *(xx, yy, yyerr, xxerr)*. If *xerr* and/or *yerr* are not defined they are not resampled and omitted from the return statement.

1.2.3 DispersionCurve

DispersionCurve class definition.

class DispersionCurve (*frequency*, *velocity*)
Bases: *swprepost.curve.Curve*

Class to define a *DispersionCurve* object.

Variables ***velocity*** (*frequency,*) – 1D array of the dispersion curve's frequency and velocity values, respectively.

__init__ (*frequency*, *velocity*)
Initialize a *DispersionCurve* object from dispersion data.

Parameters **frequency, velocity** (*iterable*) – Vector of the dispersion curve's frequency and velocity values, respectively.

Returns *DispersionCurve* – Initialized *DispersionCurve* object.

frequency

classmethod from_geopsy (*fname*)
Create from text file following the Geopsy format.

Parameters **fname** (*str*) – Name of file to be read, may be a relative or the full path.

Returns *DispersionCurve* – Instantiated *DispersionCurve* object.

slowness

txt_repr
Text representation following the Geopsy format.

velocity

wavelength

write_curve (*fileobj*)
Append *DispersionCurve* to open file object.

Parameters **fname** (*str*) – Name of file, may be a relative or the full path.

Returns *None* – Writes file to disk.

write_to_txt (*fname*, *wavetype='rayleigh'*, *mode=0*, *identifier=0*, *misfit=0.0*)
Write *DispersionCurve* to Geopsy formated file.

Parameters

- **fname** (*str*) – Name of file, may be a relative or the full path.
- **wavetype** (*{"rayleigh", "love"}, optional*) – Surface wave dispersion wavetype, default is "rayleigh".
- **mode** (*int, optional*) – Mode integer (numbered from zero), default is 0.
- **identifier** (*int, optional*) – Model identifier, default is 0.
- **misfit** (*float, optional*) – Dispersion misfit of profile, default is 0.0000.

Returns *None* – Write text representation to disk.

1.2.4 DispersionSet

DispersionSet class definition.

class DispersionSet (*identifier=0*, *misfit=0.0*, *rayleigh=None*, *love=None*)
Bases: `object`

Class for handling sets of :meth: *DispersionCurve* <*swprepost.DispersionCurve*> objects, which all belong to a common ground model.

Variables

- **love** (`rayleigh,`) – Container for *DispersionCurve* objects, of the form: *{0:DispersionCurve0, ... N:DispersionCurveN}* where each key is the mode number and the value is the corresponding instantiated *DispersionCurve* object.
- **identifier** (`int`) – Model identifier of the *DispersionSet*.

- **misfit** (*float*) – Value of dispersion misfit if provided, *None* otherwise.

__init__ (*identifier=0*, *misfit=0.0*, *rayleigh=None*, *love=None*)
Create a *DispersionCurveSet* object.

Parameters

- **identifier** (*str*) – Unique identifier of the *DispersionSet*.
- **misfit** (*float, optional*) – *DispersionSet* misfit, default is 0.0.
- **rayleigh, love** (*dict*) – Container for *DispersionCurve* objects of the form *{0:disp_curve_obj0, ... N:disp_curve_objN}* where each key is the mode number and the value is the corresponding *DispersionCurve* object.

Returns *DispersionSet* – Instantiated *DispersionSet* object.

classmethod check_type (*curveset*, *valid_type*)
Check that the *curveset* are are valid.

Specifically: 1. Assume *curveset* is instance of *dict*. 2. If it is a *dict*, check all values are instances of the *valid_type* and return zero, otherwise raise *TypeError*. 3. If it is not check if *None*, if so return one. 4. Otherwise, raise *TypeError*.

classmethod from_geopsy (*fname*, *nrayleigh='all'*, *nlove='all'*)
Create from a text file following the Geopsy format.

Parameters

- **fname** (*str*) – Name of file to be read, may be a relative or full path.
- **nrayleigh, nlove** (*{"all", int}, optional*) – Number of Rayleigh and Love modes to extract into a *DispersionSet* object, default is "all" meaning all available modes will be extracted.

Returns *DispersionSet* – Instantiated *DispersionSet* object.

write_set (*fileobj*, *nrayleigh='all'*, *nlove='all'*)
Write *DispersionSet* to current file.

Parameters **fname** (*str*) – Name of file, may be a relative or the full path.

Returns *None* – Writes file to disk.

write_to_txt (*fname*)
Write *DispersionSet* to Geopsy formated file.

Parameters **fname** (*str*) – Name of file, may be a relative or the full path.

Returns *None* – Writes text representation to disk.

1.2.5 DispersionSuite

DispersionSuite class definition.

class DispersionSuite (*dispersionset*)
Bases: *swprepost.suite.Suite*

Container for instantiated *DispersionSet* objects.

Variables ***sets*** (*list*) – Container for instantiated *DispersionSet* objects.

__init__ (*dispersionset*)
Initialize a *DispersionSuite*, from a *DispersionSet*.

Parameters **dispersionset** (*DispersionSet*) – Initialized *DispersionSet* object.

Returns *DispersionSuite* – Instantiated *DispersionSuite* object.

Raises `TypeError` – If *dispersionset* is not of type *DispersionSet*.

append(*dispersionset*, *sort=True*)
Append *DispersionSet* object to *DispersionSuite*.

Parameters **Refer to** (meth: *__init__* <*DispersionSuite.__init__*>.)

Returns *None* – Updates the attribute *sets*.

Raises `TypeError` – If *dispersionset* is not of type *DispersionSet*.

static check_input(*curveset*, *set_type*)
Check inputs comply with the required format.

Specifically: 1. *curveset* is of type *set_type*.

classmethod from_geopsy(*fname*, *nsets='all'*, *nrayleigh='all'*, *nlove='all'*, *sort=False*)
Instantiate from a text file following the Geopsy format.

Parameters

- **fname** (*str*) – Name of file, may be a relative or full path.
- **nsets** (*int, optional*) – Number of sets to extract, default is "all" so all available sets will be extracted.
- **nrayleigh, nlove** (*int, optional*) – Number of Rayleigh and Love modes respectively, default is "all" so all available modes will be extracted.
- **sort** (*bool, optional*) – Indicates whether the imported data should be sorted from lowest to highest misfit, default is *False* indicating no sorting is performed.

Returns *DispersionSuite* – Instantiated *DispersionSuite* object.

classmethod from_list(*dc_sets*, *sort=True*)
Instantiate from a list of *DispersionSet* objects.

Parameters

- **dc_sets** (*list*) – List of *DispersionSet* objects.
- **sort** (*bool, optional*) – Indicates whether the imported data should be sorted from lowest to highest misfit, default is *False* indicating no sorting is performed.

Returns *DipsersionSuite* – Instantiated *DispersionSuite* object.

sets

write_to_txt(*fname*, *nbest='all'*, *nrayleigh='all'*, *nlove='all'*)
Write to text file, following the Geopsy format.

Parameters

- **fname** (*str*) – Name of file, may be a relative or the full path.
- **nbest** (*{int, 'all'}, optional*) – Number of best models to write to file, default is 'all' indicating all models will be written.
- **nrayleigh, nlove** (*{int, 'all'}, optional*) – Number of modes to write to file, default is 'all' indicating all available modes will be written.

Returns *None* – Writes file to disk.

1.2.6 GroundModel

GroundModel class definition.

class GroundModel(*thickness*, *vp*, *vs*, *density*, *identifier=0*, *misfit=0.0*)

Bases: `object`

Class for creating and manipulating *GroundModel* objects.

Variables

- **vp, vs, rh** (*tk,*) – Thickness, compression-wave velocity (Vp), shear-wave velocity (Vs), and mass density defining each layer of the *GroundModel*, respectively.
- **identifier** (*int, optional*) – Model numeric identifier, default is 0.
- **misfit** (*float, optional*) – Model misfit, default is 0.0.

__init__(*thickness*, *vp*, *vs*, *density*, *identifier=0*, *misfit=0.0*)

Initialize a *GroundModel* object.

Parameters

- **thickness** (*iterable*) – Container of *float* or *int* denoting layer thickness (one per layer) in meters starting from the ground surface.
- **vp, vs** (*iterable*) – Container of *float* or *int* denoting the P- and S-wave velocity of each layer in m/s.
- **density** (*iterable*) – Container of *float* or *int* denoting the mass density of each layer in kg/m3.
- **identifier** (*int, optional*) – Model numeric identifier, default is 0.
- **misfit** (*float, optional*) – Model misfit, default is 0.0.

Returns *GroundModel* – Instantiated *GroundModel* object.

Raises `Various` – See :meth: *check_input_type <GroundModel.check_input_type* and :meth: *check_input_value <GroundModel.check_input_value* for details.

static calc_pr(*vp*, *vs*)

Calculate Poisson's ratio from iterable of *vp* and *vs*.

Parameters **vp, vs** (*float, int, or iterable*) – Vp and Vs values, respectively

Returns *float or list* – Poisson's ratio, calculated from each vp, vs pair.

Raises `ValueError` – If vs>vp or Poisson's ratio is negative (i.e., vp/vs too close to 1).

check_input(***kwargs*)

Check input values and types.

static check_input_type(***kwargs*)

Check inputs are of the appropriate type.

Specifically: 1. Cast *GroundModel* input to a *list* of *float*. 2. Cast meta-information (identifier and misfit to *int* and *float*).

Parameters ****kwargs** – Keyword arguments containing name and value pairs.

Raises `TypeError` – If *values* do not pass the aforementioned criteria.

static check_input_value(***kwargs*)

Check *values* have appropriate values.

Specifically: 1. Check that all *GroundModel* parameters are greater than zero. 2. Check that identifier and misfit are greater than zero. 3. Check that *vp* > *vs*.

Parameters ****kwargs** – Keyword arguments containing name and value pairs.

Raises `ValueError` – If inputs does not pass the aforementioned criteria.

depth
Return stair-step version of depth profile.

static depth_to_thick(*depths*)
Convert depths (top of each layer) to thicknesses

Parameters **depth** (*list*) – List of consecutive depths.

Returns *list* – Thickness for each layer. Half-space is defined with zero thickness.

discretize(*dmax*, *dy=0.5*, *parameter='vs'*)
Discretize a parameter of the *GroundModel*.

The *GroundModel* is discretized in terms of depth from the surface to *dmax* by *dy* such that depth will be a *list* of the form *[0:dy:dmax]*. When the discretization encounters a parameter boundary the velocity of the upper layer is assigned.

Do not use these discretized models for plotting unless *dy* is very small, as they may be misleading.

Parameters

- **dmax** (*float*) – Maximum depth of discretization.
- **dy** (*float, optional*) – Linear step of discretization in terms of depth, default is 0.5 meter.
- **parameter** (*{'vp', 'vs', 'rh', 'pr'}, optional*) – Parameter to be discretized, default is 'vs'.

Returns *Tuple* – Tuple of the form *(depth, param)* where *depth* is a *list* of the discretized depths, and *parameter* is a *list* of the discretized parameter at those depths.

Raises `KeyError` – If *parameter* is not one of those options specified.

classmethod from_geopsy(*fname*)
Create from a text file following the *Geopsy* format.

Parameters **fname** (*fname*) – File name, may contain a relative or the full path.

Returns *GroundModel* – Initialized *GroundModel* object.

Raises `Various` – If file does not follow the *Geopsy* format.

classmethod from_simple_profiles(*vp_tk*, *vp*, *vs_tk*, *vs*, *rh_tk*, *rh*)
Instantiate *GroundModel* from simple profiles.

Parameters

- **vp_tk, vs_tk, rh_tk** (*iterable*) – Iterable denoting the thicknesses of each parameter, one per layer.
- **vp, vs, rh** (*iterable*) – Iterable denoting the value of Vp, Vs, and mass density respectively.

Returns *GroundModel* – Instantiated *GroundModel* object.

gm2(*parameter*)
Parameter of *GroundModel* in stair-step form.

Parameters **parameter** (*{'depth', 'vp', 'vs', 'density', 'pr'}*) – Desired parameter to transform to stair-step profile.

Returns *list* – Defining the specified parameter.

Raises `KeyError` – If *parameter* is not one of those specified.

nlay

pr2
Return stair-step version of Poisson's ratio profile.

rh

rh2
Return stair-step version of density profile.

simplify (*parameter='vs'*)
Remove unnecessary breaks in the parameter specified.

This will typically be used for calculating the median across many profiles.

parameter [{'vs','vp','rh','density'}, optional] String denoting parameter to be simplified, default is 'vs'.

Returns *tuple* – Of the form *(thickness, parameter)* where *thickness* is a *list* of simplified thicknesses and *parameter* is a *list* of the simplified parameter.

static thick_to_depth (*thicknesses*)
Convert thickness to depth (at top of each layer).

Parameters **thickness** (*list*) – List of thicknesses defining a ground model.

Returns *list* – List of depths at the top of each layer.

tk

txt_repr
Text representation of the current *GroundModel*.

vp2
Return stair-step version of Vp profile.

vs2
Return stair-step version of Vs profile.

vs30
Calculate Vs30 of the *GroundModel*.

Vs0 is the time-averaged shear-wave velocity in the upper 30m.

Returns *float* – Vs30 of *GroundModel*.

write_model (*fileobj*)
Write model to open file object following *Geopsy* format.

Parameters **fileobj** (*_io.TextIOWrapper*) – Name of file, may be a relative or the full path.

Returns *None* – Writes file to disk.

write_to_mat (*fname_prefix*)
Save *GroundModel* information to *.mat* format.

Parameters **fname_prefix** (*str*) – Name of file (excluding the *.mat* extension) where the file should be saved, may be a relative or the full path.

Returns *None* – Writes file to disk.

write_to_txt (*fname*)
Write *GroundModel* to file that follows the *Geospy* format.

Parameters **fname** (*str*) – Name of file, may contain a relative or the full path.

Returns *None* – Writes file to disk.

1.2.7 GroundModelSuite

GroundModelSuite class definition.

class GroundModelSuite(*groundmodel*)

Bases: *swprepost.suite.Suite*

Class for manipulating suites of *GroundModel* objects.

Variables ***gms*** (*list*) – List of *GroundModel* objects, composing the suite.

__init__(*groundmodel*)

Initialize a *GroundModelSuite* from a *GroundModelObject*.

Parameters **groundmodel** (*GroundModel*) – Instantiated *GroundModel* object.

Returns *GroundModelSuite* – Initialized *GroundModelSuite*.

append(*groundmodel*, *sort=True*)

Append *GroundModel* object to *GroundModelSuite* object.

Parameters

- **groundmodel** (*GroundModel*) – refer to :meth: *__init__* <*swprepost.GroundModelSuite.__init__*>.
- **sort** (*bool*) – Sort models according to misfit (smallest to largest), default is *True* indicating sort will be performed. If it is known that the misfit of appended model is larger than those already part of the suite, setting *sort* to *False* can allow for a significant speed improvement.

Returns *None* – Instead updates the attributes *gms*.

static check_type(*groundmodel*)

Check input to *GroundModelSuite*.

Specifically: 1. *groundmodel* is of type *GroundModel*.

classmethod from_array(*tks*, *vps*, *vss*, *rhs*, *ids*, *misfits*)

Create from an array of values.

Parameters

- **tks, vps, vss, rhs** (*ndarray*) – 2D array representation of the ground models composing the suite. Each column represents a particular groundmodel and each row a layer in that ground model.
- **ids, misfits** (*ndarray*) – 1D array where each entry corresponds to a ground model.

Returns *GroundModelSuite* – Instantiated *GroundModelSuite*.

Raises `ValueError` – If the size of the arrays are inconsistent.

classmethod from_geopsy(*fname*, *nmodels='all'*, *sort=False*)

Create from a file following the *Geopsy* format.

Parameters

- **fname** (*str*) – Name of file, may contain a relative or the full path.
- **nmodels** (*{int, 'all'}, optional*) – Number of *GroundModels* to extract from file, default is *all*.

- **sort** (*bool, optional*) – Indicates whether the imported data should be sorted from lowest to highest misfit, default is *False* indicating no sorting is performed.

Returns *GroundModelSuite* – Initialized *GroundModelSuite*.

classmethod from_list(*groundmodels*, *sort=True*)
Create from a *list* of *GroundModel* objects.

gms

median(*nbest='all'*)
Calculate the median *GroundModel* of the *GroundModelSuite*.

Parameters **nbest** (*{int, 'all'}, optional*) – Number of the best profiles to consider when calculating the median profile, default is 'all', meaning all available models will be used.

Returns *GroundModel* – Initialized *GroundModel* object.

median_simple(*nbest='all'*, *parameter='vs'*)
Calculate layer-by-layer median of a given parameter.

Parameters

- **nbest** (*{int, "all"}, optional*) – Number of best models to consider, default is 'all' so all models will be used.
- **parameter** (*{'depth', 'vs', 'vp', 'rho'}, optional*) – Parameter along which to calculate the median, default is 'vs' for shear-wave velocity.

Returns *tuple* – Of the form *(median_thickness, median_parameter)* where *median_thickness* is a *list* of the median thickness of each layer and *median_parameter* is a *list* of the median parameter of each layer.

sigma_ln(*dmax=50*, *dy=0.5*, *nbest='all'*, *parameter='vs'*)
Lognormal standard deviation of a parameter.

Parameters

- **dmax** (*float, optional*) – Depth to which to discretize the parameter profiles in meters, default is 50.
- **dy** (*float, optional*) – Linear-spacing of depth samples in meters, default is 0.5.
- **nbest** (*{int, 'all'}, optional*) – Number of best profiles to consider for calculation, default is 'all'.
- **parameter** (*{'vs', 'vp', 'rh', 'density', 'pr'}, optional*) – Parameter to be used for the calculation, default is 'vs'.

Returns *Lognormal standard deviation of the nbest discretized profiles.*

vs30(*nbest='all'*)
Calculate Vs30 for *GroundModelSuite*.

Parameters **nbest** (*{int, "all"}, optional*) – Number of lowest misfit profiles to return.

Returns *list* – Of the *nbest* Vs30 values.

See also:

`Refer()`

write_to_txt(*fname*, *nbest='all'*)
Write to text file, following the Geopsy format.

Parameters

- **fname** (*str*) – Name of file, may be a relative or the full path.
- **nbest** (*{int, 'all'}, optional*) – Number of best models to write to file, default is 'all' indicating all models will be written.

Returns *None* – Writes file to disk.

1.2.8 Parameter

Parameter class definition.

class Parameter(*lay_min*, *lay_max*, *par_min*, *par_max*, *par_rev*, *lay_type='thickness'*)

Bases: `object`

Class for defining the bounds of an inversion parameter.

Variables

- **par_type** (*{'FX', 'FTL', 'LN', 'LR', 'CT', 'CD'}*) – String to denote how the layering was defined. Specifically, 'FX' = Fixed 'FTL' = Fixed Thickness Layers 'LN' = Layering by Number 'LR' = Layering Ratio 'CT' = Custom Thickness 'CD' = Custom Depth
- **par_add_value** (*par_value,*) – Numerical values to provide context about the type of layering selected. *par_add_value* only used for three letter parameter types (i.e., 'FTL').
- **lay_max** (*lay_min,*) – Minimum and maximum thickness or depth of each layer, respectively.
- **par_max** (*par_min,*) – Minimum and maximum potential value of the parameter, one *float* value per layer.
- **par_rev** (*list*) – Indicate whether to allow parameter reversals, one *bool* value per layer.

__init__(*lay_min*, *lay_max*, *par_min*, *par_max*, *par_rev*, *lay_type='thickness'*)

Initialize a *Parameter* object.

Parameters

- **lay_min, lay_max** (*iterable*) – Minimum and maximum thickness or depth of each layer, respectively.
- **par_min, par_max** (*iterable*) – Minimum and maximum potential value of the parameter, one *float* per layer.
- **par_rev** (*iterable*) – Indicate whether to allow parameter reversals, one *bool* per layer.
- **lay_type** (*{'thickness', 'depth'}, optional*) – Indicate whether the layers are defined in terms of depth or thickness.

Returns *Parameter* – Instantiated *Parameter* object.

static check_depth_factor(*depth_factor*)

Check input value for factor.

static check_layers(*lower_name*, *lower*, *upper_name*, *upper*)

Check layering input.

Specifically:

1. Check that *lower* and *upper* are the same length.
2. Ensure that each value for *lower* is less than the corresponding value for *upper*.

static check_rev(*par_rev*)
Check reversal input.

Specifically:

1. *par_rev* is a list of ‘bool‘s.

classmethod clone(*parameter*)
Copy provided *Parameter* object.

static depth_ftl(*nlayers*, *thickness*)
Calculate min and max thicknesses for each layer using FTL.

Parameters

- **nlayers** (*int*) – Desired number of layers.
- **thickness** (*float*) – Thickness of each layer.

Returns *Tuple* – Tuple of lists indicating thicknesses of the form ([minthickness...], [maxthickness...]).

static depth_ln(*wmin*, *wmax*, *nlayers*, *depth_factor=2*)
Calculate min and max depth for each layer using LN.

Parameters

- **wmin, wmax** (*float*) – Minimum and maximum measured wavelength from the fundamental mode Rayleigh wave disperison.
- **nlayers** (*int*) – Desired number of layers.
- **depth_factor** (*[float, int], optional*) – Factor by which the maximum wavelength is divided to estimate the maximum depth of profiling, default is 2.

Returns *Tuple* – Tuple of lists indicating depths of the form ([mindepth...], [maxdepth...]).

static depth_lr(*wmin*, *wmax*, *lr*, *depth_factor=2*)
Return minimum and maximum depth for each layer using the Layering Ratio approach developed by Cox and Teague (2016).

Note that the Layering Ratio approach implemented here has been modified slightly to ensure the maximum depth of the last layer does not exceed dmax. Suggestions for solving this issue are hinted at in Cox and Teague (2016), but not provided explicitly.

Parameters

- **wmin, wmax** (*float*) – Minimum and maximum measured wavelength from the fundamental mode Rayleigh wave dispersion.
- **lr** (*float*) – Layering Ratio, this controls the number of layers and their potential thicknesses, refer to Cox and Teague 2016 for details.
- **depth_factor** (*[float, int], optional*) – Factor by which the maximum wavelength is divided to estimate the maximum depth of profiling, default is 2.

Returns *Tuple* – Tuple of lists indicating depths of the form ([mindepth...], [maxdepth...]).

classmethod from_ftl(*nlayers*, *thickness*, *par_min*, *par_max*, *par_rev=False*)
Create *Parameter* using Fixed Thickness Layering (FTL).

Parameters

- **nlayers** (*int*) – Number of desired layers.
- **thickness** (*float*) – Thickness of all layers in meters.

- **par_min, par_max** (*float*) – Minimum and maximum potential value of the parameter, applied to all layers.
- **par_rev** (*bool, optional*) – Indicate whether layers are allowed to reverse or not, default is *False* (i.e., no reversal allowed).

Returns *Parameter* – Instantiated *Parameter* object.

Note: If a more detailed parameterization is desired than available here use the *dinver* user inferface to tweak the resulting *.param* file.

---

classmethod from_fx(*value*)
Create *Parameter* using Fixed (FX) layering.

Parameters value (*{float, int}*) – Value assigned to the parameter at all depths. Value will not be allowed to change with depth.

Returns *Parameter* – Instantiated *Parameter* object.

classmethod from_ln(*wmin*, *wmax*, *nlayers*, *par_min*, *par_max*, *par_rev*, *depth_factor=2*)
Create *Parameter* using Layering by Number (LN).

Parameters

- **wmin, wmax** (*float*) – Minimum and maximum measured wavelength from the fundamental mode Rayleigh wave disperison.
- **nlayers** (*int*) – Desired number of layers.
- **par_min, par_max** (*float*) – Minimum and maximum potential value of the parameter, applied to all layers.
- **par_rev** (*bool, optional*) – Indicate whether layers are allowed to reverse or not, default is *False* (i.e., no reversal allowed).
- **depth_factor** (*[float, int], optional*) – Factor by which the maximum wavelength is divided to estimate the maximum depth of profiling, default is 2.

Returns *Parameter* – Instantiated *Parameter* object.

classmethod from_lr(*wmin*, *wmax*, *lr*, *par_min*, *par_max*, *par_rev*, *depth_factor=2*)
Alternate constructor to instantiate a *Parameter* using LR.

Use Layering Ratio (LR) to define the *Parameter*.

Parameters

- **wmin, wmax** (*float*) – Minimum and maximum measured wavelength from the fundamental mode Rayleigh wave dispersion.
- **lr** (*float*) – Layering Ratio, this controls the number of layers and their potential thicknesses, refer to Cox and Teague (2016) for details.
- **par_min, par_max** (*float*) – Minimum and maximum potential value of the parameter, applied to all layers.
- **par_rev** (*bool, optional*) – Indicate whether layers are allowed to reverse or not, default is *False* (i.e., no reversal allowed).
- **depth_factor** (*[float, int], optional*) – Factor by which the maximum wavelength is divided to estimate the maximum depth of profiling, default is 2.

Returns *Parameter* – Instantiated *Parameter* object.

Note: If a more detailed parameterization is desired than available here use the *dinver* user inferface to tweak the resulting *.param* file.

classmethod from_parameter_and_link(*par_min*, *par_max*, *par_rev*, *existing_parameter*, *ptype='vs'*)

Create *Parameter* from an existing *Parameter* and link.

Parameters

- **par_min, par_max** (*float*) – Minimum and maximum potential value of the parameter, applied to all layers.
- **par_rev** (*bool, optional*) – Indicate whether layers are allowed to reverse or not, default is *False* (i.e., no reversal allowed).
- **existing_parameter** (*Parameter*) – Instantiated *Parameter* object to which you wish to link the current parameter.
- **ptype** (*{'vs', 'pr', 'rh', 'vp'}, optional*) – Inversion parameter, represented by the *existing_parameter*, default is *vs*.

Returns *Parameter* – Instantiated *Parameter* with the same layering as *existing_parameter*, but different bounds.

Note: If a more detailed parameterization is desired than available here use the *dinver* user inferface to tweak the resulting *.param* file.

lay_type

static make_rectangle(*left*, *right*, *upper*, *lower*)

static min_max_rev(*nlayers*, *par_min*, *par_max*, *par_rev*)

Create list for par_min, par_max, par_rev.

plot(*ax=None*, *show_example=True*)

1.2.9 Parameterization

Parameterization class definition.

class Parameterization(*vp*, *pr*, *vs*, *rh*)

Bases: `object`

Class for developing inversion parameterizations.

This class is intended to be used for developing various parameterization files for use in the open-source software Dinver. While parameterizations of various kinds can be built quickly using this tool, it does have limited functionality compared to the full user interface. It is recommended that the you use this tool to create general parameterizations in batches and fine tune them if necessary using the Dinver user interface.

Variables **vp, pr, *rh*** (*vs,*) – Parameter objects defining shear-wave velocity, compression-wave velocity, Poisson's ratio, and mass density respectively.

__init__(*vp*, *pr*, *vs*, *rh*)

Initialize an instance of the *Parameterization* class.

Initialize a *Parameterization* using instantiated *Parameter* objects.

Parameters **vp, pr, vs, rh** (*Parameter*) – Instantiated *Parameter* objects, see :meth: *Parameter <swprepost.Parameter.__init__>*.

Returns *Parameterization* – An initialized *Parameterization* object.

Raises `TypeError` – If *vp*, *pr*, *vs*, and *rh* are not *Parameter* objects.

static check_parameter(*name*, *val*)
Check input for the parameter.

classmethod from_min_max(**args*, ***kwargs*)

classmethod from_param(*fname_prefix*, *version='3.4.2'*)
Create *Parameterization* from a .param file.

Note this method is still largely experimental and may not work for all cases.

Parameters

- **fname_prefix** (*str*) – Name of param file to be opened excluding the *.param* suffix, may include the relative or full path.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *Parameterization* – Instantiated *Parameterization* object.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

to_param(*fname_prefix*, *version='3.4.2'*)
Write info to the .param used by Dinver.

Parameters

- **fname_prefix** (*str*) – File name prefix (without the .param extension), may be a relative or the full path.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *None* – Writes *.param* file to disk.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

1.2.10 Suite

Suite class definition.

class Suite(*item*)
Bases: abc.ABC

__init__(*item*)
Create *Suite* from *item*.

identifiers

misfit_range(*nmodels='all'*)
Return range of misfits for nmodels.

Parameters **nmodels** (*{int, "all"}, optional*) – Number of models to consider, default is 'all' so all available models will be considered.

Returns *float, tuple* – If *nmodels==1*, returns *float* corresponding to the single best misfit, otherwise returns *tuple* of the form (min_msft, max_msft).

misfit_repr(*nmodels='all'*, ***kwargs*)
String representation of misfit [min-max] or [min].

Parameters

- **nmodels** (*{int, "all"}, optional*) – Number of models to consider, default is 'all' so all available models will be considered.
- ****kwargs** – Optional keyword arguments for *np.format_float_positional* https://docs.scipy.org/doc/numpy-1.15.1/reference/generated/numpy.format_float_positional.html

Returns *str* – Representation of the misfit values for the selected suite.

misfits

size

1.2.11 ModalTarget

Definition of ModalTarget class.

class ModalTarget(*frequency*, *velocity*, *velstd*, *description=(('rayleigh', 0),)*)
Bases: *swprepost.curveuncertain.CurveUncertain*

Target information for a surface wave mode.

ModalTarget is a class for loading, manipulating, and writting surface wave target information for a single mode in preparation for surface wave inversion.

__init__(*frequency*, *velocity*, *velstd*, *description=(('rayleigh', 0),)*)
Initialize a *ModalTarget* object.

Parameters

- **frequency, velocity, velstd** (*array-like*) – Arrays of frequency, velocity, and velocity standard deviation values to fully describe a mode of experimental dispersion data (one per point).
- **description** (*tuple of tuples*) – Each *ModalTarget* may describe one or more wavetypes and/or one or more modes. Each potential description of the ModalTarget is listed as tuple of the form *(wavetype, modenumber)* where *wavetype* is either "rayleigh" or "love" and *modenumber* is a non-negative integer. A mode number of zero refers to the fundamental

mode. The potential descriptions of a mode are grouped into a tuple containing all possible descriptions. The default description is that of the fundamental Rayleigh mode expressed as *(("rayleigh", 0),)*.

Returns *ModalTarget* – Instantiated *ModalTarget* object.

Raises

- `TypeError` – If *frequency*, *velocity*, and *velstd* are not *array-like*.
- `ValueError` – If *velstd* is *float* and the value is less than zero.

cov

cut (*pmin*, *pmax*, *domain='frequency'*)
Remove data outside of the specified range.

Parameters

- **pmin, pmax** (*float*) – New minimum and maximum parameter value in the specified domain, respectively.
- **domain** (*{'frequency', 'wavelength'}, optional*) – Domain along which to perform the cut.

Returns *None* – May update attributes *frequency*, *velocity*, and *velstd*.

easy_resample (*pmin*, *pmax*, *pn*, *res_type='log'*, *domain='wavelength'*, *inplace=False*)
Resample dispersion curve.

Resample dispersion curve over a specific range, using log or linear sampling in the frequency or wavelength domain.

Parameters

- **pmin, pmax** (*float*) – Minimum and maximum parameter value in the resampled dispersion data.
- **pn** (*int*) – Number of points in the resampled dispersion data.
- **res_type** (*{'log', 'linear'}, optional*) – Resample using either logarithmic or linear sampling, default is logarithmic.
- **domain** (*{'frequency', 'wavelength'}, optional*) – Domain along which to perform the resampling.
- **inplace** (*bool*) – Indicating whether the resampling should be done in place or if a new *Target* object should be returned.

Returns *None or Target* – If *inplace=True* returns *None*, and may update attributes *frequency*, *velocity*, and *velstd*. If *inplace=False* a new *Target* object is returned.

Raises `NotImplementedError` – If *res_type* and/or *domain* are not among the options specified.

frequency

classmethod from_csv (*fname*, *description=(('rayleigh', 0),)*)
Read *ModalTarget* from csv.

Read a comma seperated values (csv) file with header line(s) to construct a *ModalTarget*.

Parameters

- **fname** (*str*) – Relative or the full path to a file containing surface wave dispersion data. The field should have three columns: frequency in Hz, velocity in m/s, and velocity standard deviation in m/s.

- **description** (*tuple of tuples*) – Each *ModalTarget* may describe one or more wavetypes and/or one or more modes. Each potential description of the ModalTarget is listed as tuple of the form *(wavetype, modenumber)* where *wavetype* is either "rayleigh" or "love" and *modenumber* is a non-negative integer. A mode number of zero refers to the fundamental mode. The potential descriptions of a mode are grouped into a tuple containing all possible descriptions. The default description is that of the fundamental Rayleigh mode expressed as *(("rayleigh", 0),)*.

Returns *ModalTarget* – Initialized *ModalTarget* object.

Raises `ValueError` – If the format of the input file does not match that detailed above.

classmethod from_target(*fname_prefix*, *version='3.4.2'*)

Create from target file.

Note that this method is still largely experimental and may not work for all cases.

Parameters

- **fname_prefix** (*str*) – Name of target file to be opened excluding the *.target* suffix, may include the relative or full path.
- **version** (*{'2.10.1', '3.10.2'}, optional*) – Major version of Geopsy that was used to write the target file, default is '3.4.2'.

Returns *ModalTarget* – Instantiated *ModalTarget* object.

classmethod from_txt_dinver(*fname*, *version='3.4.2'*)

Create from text format accepted by *Dinver*.

Parameters

- **fname** (*str*) – Name of output file, may a relative or full path.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *ModalTarget* – Instantiated *ModalTarget* with information from file.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

classmethod from_wavelength(*wavelength*, *velocity*, *velstd*, *description=(('rayleigh', 0),)*)

Create from data processed in terms of wavelength.

Parameters

- **frequency, velocity, velstd** (*array-like*) – Arrays of frequency, velocity, and velocity standard deviation values to fully describe a mode of experimental dispersion data (one per point).
- **description** (*tuple of tuples*) – Each *ModalTarget* may describe one or more wavetypes and/or one or more modes. Each potential description of the ModalTarget is listed as tuple of the form *(wavetype, modenumber)* where *wavetype* is either "rayleigh" or "love" and *modenumber* is a non-negative integer. A mode number of zero refers to the fundamental mode. The potential descriptions of a mode are grouped into a tuple containing all possible

descriptions. The default description is that of the fundamental Rayleigh mode expressed as *(("rayleigh", 0),)*.

Returns *ModalTarget* – Instantiated *ModalTarget* object.

is_no_velstd
Indicates *True* if every point has zero *velstd*.

logstd
Get logarithmic slowness standard deviation.

plot(*x='frequency'*, *y='velocity'*, *yerr='velstd'*, *ax=None*, *figkwargs=None*, *errorbarkwargs=None*)
Plot *ModalTarget* information.

Parameters

- **x** (*{'frequency', 'wavelength'}, optional*) – Select what should be plotted along the x-axis, default is 'frequency'.
- **y** (*{'velocity', 'slowness'}, optional*) – Select what should be plotted along the y-axis, default is 'velocity'.
- **yerr** (*{'velstd', 'slostd'}, optional*) – Select what should be plotted as the y-error, default is 'velstd'.
- **ax** (*Axis, optional*) – Provide an axes on which to plot, default is *None* meaning an axes will be created on-the-fly.
- **figkwargs** (*dict*) – Additional keyword arguments defining the *Figure*. Ignored if *ax* is defined.
- **errorbarkwargs** (*dict*) – Additional keyword arguments defining the styling of the error-bar plot.

Returns *None or Tuple* – If *ax* is defined this method returns returns *None*. If *ax=None* this method returns a *tuple* of the form *(fig, ax)* where *fig* is a *Figure* object and *ax* is an *Axes* object.

pseudo_depth(*depth_factor=2.5*)
Estimate depth based on the experimental dispersion data.

This method, along with :meth: *pseudo-vs*, may be useful to create plots of pseudo-Vs vs pseudo-depth for selecting appropriate boundaries for parameter limits in the inversion stage.

Parameters **depth_factor** (*float, optional*) – Factor by which the mean wavelegnth is divided to produce an estimate of depth. Typical are between 2 and 3, default 2.5.

Returns *ndarray* – Of pseudo-depth.

pseudo_vs(*velocity_factor=1.1*)
Estimate Vs based on the experimental dispersion data.

This method, along with :meth: *pseudo-depth*, may be useful to create plots of pseudo-Vs vs pseudo-depth for selecting appropriate boundaries for parameter limits.

Parameters **velocity_factor** (*float, optional*) – Factor by which the mean Rayleigh wave velocity is multiplied to produce an estimate of shear-wave velocity. Typically range between 1 and 1.2, and is dependent upon the expected Poisson's ratio, default is 1.1.

Returns *ndarray* – Of pseudo-vs.

setcov(*cov*)
Set coefficient of variation (COV) to a constant value.

This method may be used if no velocity standard deviation was measured or provided. In general, a COV between 0.05 and 0.10 should provide a reasonable estimate of the uncertainty.

If velocity standard deviations have already been provided this method will overwrite them. If this is not desired refer to :meth: *setmincov*.

Parameters **cov** (*float*) – Coefficient of variation to be used to replace *velstd*.

Returns *None* – Updates attribute *velstd*.

Raises ValueError: – If *cov* < 0.

setmincov (*cov*)
Set minimum coefficient of variation (COV).

If uncertainty in the experimental data has been provided, this method allows the setting of a minimum COV, where all data points with uncertainty below this threshold will be modified and those above this threshold will be left alone.

If no measure of uncertainty has been provided, prefer :meth: *setcov*.

Parameters **cov** (*float*) – Minimum allowable COV.

Returns *None* – May update attribute *velstd*.

Raises `ValueError` – If *cov* < 0.

slostd
Get slowness standard deviation.

slowness

to_csv (*fname*)
Write *ModalTarget* to csv.

Parameters **fname** (*str*) – Name of output file, may a relative or full path.

Returns *None* – Writes file to disk.

to_target (*fname_prefix*, *version='3.4.2'*)
Write info to the .target file format used by *Dinver*.

Parameters

- **fname_prefix** (*str*) – Name of target file without the .target suffix, a relative or full path may be provided.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *None* – Writes file to disk.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

to_txt_dinver (*fname*, *version='3.4.2'*)
Write in text format accepted by *Dinver*.

Parameters

- **fname** (*str*) – Name of output file, may a relative or full path.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *None* – Write's geopsy-styled text file to disk.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

velocity

velstd

vr40
Estimate Rayleigh wave velocity at a wavelength of 40m.

wavelength

Target
alias of *swprepost.modaltarget.ModalTarget*

1.2.12 TargetSet

Definition of TargetSet class.

class TargetSet(*targets: List[swprepost.modaltarget.ModalTarget]*)
Bases: `object`

Container for handling multiple inversion targets.

__init__(*targets: List[swprepost.modaltarget.ModalTarget]*) → None
Initialize a *TargetSet* object.

Parameters **targets** (*list*) – List of *ModalTargets* that define *TargetSet*.

cut(*pmin*, *pmax*, *domain='frequency'*)
Remove data outside of the specified range.

Parameters

- **pmin, pmax** (*float*) – New minimum and maximum parameter value in the specified domain, respectively.
- **domain** (*{'frequency', 'wavelength'}, optional*) – Domain along which to perform the cut.

Returns *None* – May update attributes *frequency*, *velocity*, and *velstd*.

easy_resample(*pmin*, *pmax*, *pn*, *res_type='log'*, *domain='wavelength'*, *inplace=False*)
Resample dispersion curve.

Resample dispersion curve over a specific range, using log or linear sampling in the frequency or wavelength domain.

Parameters

- **pmin, pmax** (*float*) – Minimum and maximum parameter value in the resampled dispersion data.
- **pn** (*int*) – Number of points in the resampled dispersion data.
- **res_type** (*{'log', 'linear'}, optional*) – Resample using either logarithmic or linear sampling, default is logarithmic.
- **domain** (*{'frequency', 'wavelength'}, optional*) – Domain along which to perform the resampling.
- **inplace** (*bool*) – Indicating whether the resampling should be done in place or if a new *Target* object should be returned.

Returns *None or Target* – If *inplace=True* returns *None*, and may update attributes *frequency*, *velocity*, and *velstd*. If *inplace=False* a new *Target* object is returned.

Raises `NotImplementedError` – If *res_type* and/or *domain* are not among the options specified.

classmethod from_file (*fname*, *file_format=None*, *version='3.4.2'*)
Read *TargetSet* info from disk.

Parameters

- **fname** (*str*) – Name of file, a relative or full path may be provided.
- **file_format** (*{'.target'}, optional*) – Format of the file, default is *None* indicating extension of fname will be used.

Returns *TargetSet* – Created from information stored in the provided file.

Raises `ValueError` – If *fname* does not have an extension and *format* is *None*.

classmethod from_target (*fname_prefix*, *version='3.4.2'*)
Create *TargetSet* from .target file.

Note this method is still largely experimental and may not work for all cases.

Parameters

- **fname_prefix** (*str*) – Name of target file to be opened excluding the *.target* suffix, may include the relative or full path.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *TargetSet* – Instantiated *TargetSet* object.

Raises `NotImplementedError` – If *version* does not match the options provided.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

to_file (*fname*, *file_format=None*, *version='3.4.2'*)
Write *TargetSet* info to disk.

Parameters

- **fname** (*str*) – Name of file, a relative or full path may be provided.

- **file_format** (*{'.target'}, optional*) – Format of the file, default is *None* indicating extension of fname will be used.

Returns *None* – Writes *TargetSet* object to disk.

Raises `ValueError` – If *fname* does not have an extension and *file_format* is *None*.

to_target (*fname_prefix*, *version='3.4.2'*)
Write info to the .target file format used by Dinver.

Parameters

- **fname_prefix** (*str*) – Name of target file without the .target suffix, a relative or full path may be provided.
- **version** (*{'3.4.2', '2.10.1'}, optional*) – Version of Geopsy, default is version '3.4.2'.

Returns *None* – Writes *.target* file to disk.

Notes

In previous versions of *swprepost* (v1.0.0 and earlier) an attempt was made to support all versions of Dinver's .target and .param formats. However, this has become untenable due to the number and frequency of breaking changes that occur to these formats. Therefore, in lieu of supporting all versions, *swprepost* will seek to support only those versions directly associated with the open-source high-performance computing application *swbatch*.

1.3 License Information

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

Symbols

A

C

D

E

F

G

I

L

M

N

P

R

S

T

V

W