Welcome to powerlaw's documentation!¶

Here are documentation for the functions and classes in powerlaw. See the powerlaw home page for more information and examples.

Contents:

class `powerlaw.Distribution`(*xmin=1*, *xmax=None*, *discrete=False*, *fit_method='Likelihood'*, *data=None*, *parameters=None*, *parameter_range=None*, *initial_parameters=None*, *discrete_approximation='round'*, *parent_Fit=None*, ***kwargs*)[source]¶

An abstract class for theoretical probability distributions. Can be created with particular parameter values, or fitted to a dataset. Fitting is by maximum likelihood estimation by default.

Parameters: **xmin** : int or float, optional

The data value beyond which distributions should be fitted. If None an optimal one will be calculated.

xmax : int or float, optional

The maximum value of the fitted distributions.

discrete : boolean, optional

Whether the distribution is discrete (integers).

data : list or array, optional

The data to which to fit the distribution. If provided, the fit will be created at initialization.

fit_method : "Likelihood" or "KS", optional

Method for fitting the distribution. "Likelihood" is maximum Likelihood estimation. "KS" is minimial distance estimation using The Kolmogorov-Smirnov test.

parameters : tuple or list, optional

The parameters of the distribution. Will be overridden if data is given or the fit method is called.

parameter_range : dict, optional

Dictionary of valid parameter ranges for fitting. Formatted as a dictionary of parameter names ('alpha' and/or 'sigma') and tuples of their lower and upper limits (ex. (1.5, 2.5), (None, .1)

initial_parameters : tuple or list, optional

Initial values for the parameter in the fitting search.

discrete_approximation : "round", "xmax" or int, optional

If the discrete form of the theoeretical distribution is not known, it can be estimated. One estimation method is "round", which sums the

probability mass from x-.5 to x+.5 for each data point. The other option is to calculate the probability for each x from 1 to N and normalize by their sum. N can be “xmax” or an integer.

parent_Fit : Fit object, optional

A Fit object from which to use data, if it exists.

Methods

KS([data])	Returns the Kolmogorov-Smirnov distance D between the distribution and the data.
ccdf([data, survival])	The complementary cumulative distribution function (CCDF) of the theoretical distribution.
cdf([data, survival])	The cumulative distribution function (CDF) of the theoretical distribution.
fit([data, suppress_output])	Fits the parameters of the distribution to the data.
generate_random([n, estimate_discrete])	Generates random numbers from the theoretical probability distribution.
in_range()	Whether the current parameters of the distribution are within the range of valid parameters.
initial_parameters(data)	Return previously user-provided initial parameters or, if never provided, calculate new ones.
likelihoods(data)	The likelihoods of the observed data from the theoretical distribution.
loglikelihoods(data)	The logarithm of the likelihoods of the observed data from

	the theoretical distribution.
parameter_range(r[, initial_parameters])	Set the limits on the range of valid parameters to be considered while fitting.
pdf([data])	Returns the probability density function (normalized histogram) of the theoretical distribution for the values in data within xmin and xmax, if present.
plot_ccdf([data, ax, survival])	Plots the complementary cumulative distribution function (CDF) of the theoretical distribution for the values given in data within xmin and xmax, if present.
plot_cdf([data, ax, survival])	Plots the cumulative distribution function (CDF) of the theoretical distribution for the values given in data within xmin and xmax, if present.
plot_pdf([data, ax])	Plots the probability density function (PDF) of the theoretical distribution for the values given in data within xmin and xmax, if present.

KS(*data=None*)[source]¶

Returns the Kolmogorov-Smirnov distance D between the distribution and the data. Also sets the properties D+, D-, V (the Kuiper testing statistic), and Kappa (1 + the average difference between the theoretical and empirical distributions).

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

`ccdf`(*data=None*, *survival=True*)[source]¶

The complementary cumulative distribution function (CCDF) of the theoretical distribution. Calculated for the values given in data within xmin and xmax, if present.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

survival : bool, optional

Whether to calculate a CDF (False) or CCDF (True). True by default.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`cdf`(*data=None*, *survival=False*)[source]¶

The cumulative distribution function (CDF) of the theoretical distribution. Calculated for the values given in data within xmin and xmax, if present.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

survival : bool, optional

Whether to calculate a CDF (False) or CCDF (True). False by default.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`fit`(*data=None*, *suppress_output=False*)[source]¶

Fits the parameters of the distribution to the data. Uses options set at initialization.

`generate_random`(*n=1*, *estimate_discrete=None*)[source]¶

Generates random numbers from the theoretical probability distribution. If xmax is present, it is currently ignored.

Parameters: **n** : int or float

The number of random numbers to generate

estimate_discrete : boolean

For discrete distributions, whether to use a faster approximation of the random number generator. If None, attempts to inherit the estimate_discrete behavior used for fitting from the Distribution object or the parent Fit object, if present. Approximations only exist for some distributions (namely the power law). If an approximation does not exist an estimate_discrete setting of True will not be inherited.

Returns: **r** : array

Random numbers drawn from the distribution

`in_range`()[source]¶

Whether the current parameters of the distribution are within the range of valid parameters.

`initial_parameters`(*data*)[source]¶

Return previously user-provided initial parameters or, if never provided, calculate new ones. Default initial parameter estimates are unique to each theoretical distribution.

`likelihoods`(*data*)[source]¶

The likelihoods of the observed data from the theoretical distribution. Another name for the probabilities or probability density function.

`loglikelihoods`(*data*)[source]¶

The logarithm of the likelihoods of the observed data from the theoretical distribution.

`parameter_range`(*r*, *initial_parameters=None*)[source]¶

Set the limits on the range of valid parameters to be considered while fitting.

Parameters: **r** : dict

A dictionary of the parameter range. Restricted parameter names are keys, and with tuples of the form (lower_bound, upper_bound) as values.

initial_parameters : tuple or list, optional

Initial parameter values to start the fitting search from.

`pdf`(*data=None*)[source]¶

Returns the probability density function (normalized histogram) of the theoretical distribution for the values in data within xmin and xmax, if present.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in

which the Distribution object is contained.

Returns: **probabilities** : array

`plot_ccdf`(*data=None*, *ax=None*, *survival=True*, ***kwargs*)[source]¶

Plots the complementary cumulative distribution function (CDF) of the theoretical distribution for the values given in data within xmin and xmax, if present. Plots to a new figure or to axis ax if provided.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

ax : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

survival : bool, optional

Whether to plot a CDF (False) or CCDF (True). True by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`plot_cdf`(*data=None*, *ax=None*, *survival=False*, ***kwargs*)[source]¶

Plots the cumulative distribution function (CDF) of the theoretical distribution for the values given in data within xmin and xmax, if present. Plots to a new figure or to axis ax if provided.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

ax : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

survival : bool, optional

Whether to plot a CDF (False) or CCDF (True). False by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`plot_pdf`(*data=None*, *ax=None*, ***kwargs*)[source]¶

Plots the probability density function (PDF) of the theoretical distribution for the values given in data within xmin and xmax, if present. Plots to a new figure or to axis ax if provided.

Parameters: **data** : list or array, optional

If not provided, attempts to use the data from the Fit object in which the Distribution object is contained.

ax : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

class powerlaw.Fit(*data*, *discrete=False*, *xmin=None*, *xmax=None*, *verbose=True*, *fit_method='Likelihood'*, *estimate_discrete=True*, *discrete_approximation='round'*, *sigma_threshold=None*, *parameter_range=None*, *fit_optimizer=None*, *xmin_distance='D'*, ***kwargs*)[source]¶

A fit of a data set to various probability distributions, namely power laws. For fits to power laws, the methods of Clauset et al. 2007 are used. These methods identify the portion of the tail of the distribution that follows a power law, beyond a value xmin. If no xmin is provided, the optimal one is calculated and assigned at initialization.

Parameters: **data** : list or array

discrete : boolean, optional

Whether the data is discrete (integers).

xmin : int or float, optional

The data value beyond which distributions should be fitted. If None an optimal one will be calculated.

xmax : int or float, optional

The maximum value of the fitted distributions.

verbose: bool, optional

Whether to print updates about where we are in the fitting process. Default True.

estimate_discrete : bool, optional

Whether to estimate the fit of a discrete power law using fast analytical methods, instead of calculating the fit exactly with slow numerical methods. Very accurate with xmin>6

sigma_threshold : float, optional

Upper limit on the standard error of the power law fit. Used after fitting, when identifying valid xmin values.

parameter_range : dict, optional

Dictionary of valid parameter ranges for fitting. Formatted as a dictionary of parameter names (‘alpha’ and/or ‘sigma’) and tuples of

their lower and upper limits (ex. (1.5, 2.5), (None, .1)

Methods

ccdf([original_data, survival])	Returns the complementary cumulative distribution function of the data.
cdf([original_data, survival])	Returns the cumulative distribution function of the data.
distribution_compare(dist1, dist2[, nested])	Returns the loglikelihood ratio, and its p-value, between the two distribution fits, assuming the candidate distributions are nested.
find_xmin([xmin_distance])	Returns the optimal xmin beyond which the scaling regime of the power law fits best.
loglikelihood_ratio(dist1, dist2[, nested])	Another name for distribution_compare.
nested_distribution_compare(dist1, dist2[, ...])	Returns the loglikelihood ratio, and its p-value, between the two distribution fits, assuming the candidate distributions are nested.
pdf([original_data])	Returns the probability density function (normalized histogram) of the data.

plot_ccdf([ax, original_data, survival])	Plots the CCDF to a new figure or to axis ax if provided.
plot_cdf([ax, original_data, survival])	Plots the CDF to a new figure or to axis ax if provided.
plot_pdf([ax, original_data, linear_bins])	Plots the probability density function (PDF) or the data to a new figure or to axis ax if provided.

ccdf(*original_data=False*, *survival=True*, ***kwargs*)[source]¶

Returns the complementary cumulative distribution function of the data.

Parameters: **original_data** : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

survival : bool, optional

Whether to return the complementary cumulative distribution function, also known as the survival function, or the cumulative distribution function, 1-CCDF.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is greater than or equal to X.

cdf(*original_data=False*, *survival=False*, ***kwargs*)[source]¶

Returns the cumulative distribution function of the data.

Parameters: **original_data** : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

survival : bool, optional

Whether to return the complementary cumulative distribution function, 1-CDF, also known as the survival function.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`distribution_compare`(*dist1*, *dist2*, *nested=None*, ***kwargs*)[source]¶

Returns the loglikelihood ratio, and its p-value, between the two distribution fits, assuming the candidate distributions are nested.

Parameters: dist1 : string

Name of the first candidate distribution (ex. ‘power_law’)

dist2 : string

Name of the second candidate distribution (ex. ‘exponential’)

nested : bool or None, optional

Whether to assume the candidate distributions are nested versions of each other. None assumes not unless the name of one distribution is a substring of the other.

Returns: R : float

Loglikelihood ratio of the two distributions’ fit to the data. If greater than 0, the first distribution is preferred. If less than 0, the second distribution is preferred.

p: float

Significance of R

`find_xmin`(*xmin_distance=None*)[source]¶

Returns the optimal xmin beyond which the scaling regime of the power law fits best. The attribute self.xmin of the Fit object is also set.

The optimal xmin beyond which the scaling regime of the power law fits best is identified by minimizing the Kolmogorov-Smirnov distance between the data and the theoretical power law fit. This is the method of Clauset et al. 2007.

`loglikelihood_ratio`(*dist1*, *dist2*, *nested=None*, ***kwargs*)[source]¶

Another name for distribution_compare.

`nested_distribution_compare`(*dist1*, *dist2*, *nested=True*, ***kwargs*)[source]¶

Returns the loglikelihood ratio, and its p-value, between the two distribution fits, assuming the candidate distributions are nested.

Parameters: dist1 : string

Name of the first candidate distribution (ex. ‘power_law’)

dist2 : string

Name of the second candidate distribution (ex. ‘exponential’)

nested : bool or None, optional

Whether to assume the candidate distributions are nested versions of each other. None assumes not unless the name of one distribution is a substring of the other. True by default.

Returns: **R** : float

Loglikelihood ratio of the two distributions’ fit to the data. If greater than 0, the first distribution is preferred. If less than 0, the second distribution is preferred.

p : float

Significance of R

`pdf`(*original_data=False*, ***kwargs*)[source]¶

Returns the probability density function (normalized histogram) of the data.

Parameters: **original_data** : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

Returns: **bin_edges** : array

The edges of the bins of the probability density function.

probabilities : array

The portion of the data that is within the bin. Length 1 less than bin_edges, as it corresponds to the spaces between them.

`plot_ccdf`(*ax=None*, *original_data=False*, *survival=True*, ***kwargs*)[source]¶

Plots the CCDF to a new figure or to axis ax if provided.

Parameters: **ax** : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

original_data : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

survival : bool, optional

Whether to plot a CDF (False) or CCDF (True). True by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`plot_cdf`(*ax=None*, *original_data=False*, *survival=False*, ***kwargs*)[source]¶

Plots the CDF to a new figure or to axis ax if provided.

Parameters: **ax** : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

original_data : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

survival : bool, optional

Whether to plot a CDF (False) or CCDF (True). False by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`plot_pdf`(*ax=None*, *original_data=False*, *linear_bins=False*, ***kwargs*)[source]¶

Plots the probability density function (PDF) or the data to a new figure or to axis ax if provided.

Parameters: **ax** : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

original_data : bool, optional

Whether to use all of the data initially passed to the Fit object. If False, uses only the data used for the fit (within xmin and xmax.)

linear_bins : bool, optional

Whether to use linearly spaced bins (True) or logarithmically spaced bins (False). False by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`powerlaw.bisect_map`(*mn*, *mx*, *function*, *target*)[source]¶

Uses binary search to find the target solution to a function, searching in a given ordered sequence of integer values.

Parameters: **seq** : list or array, monotonically increasing integers

function : a function that takes a single integer input, which monotonically

decreases over the range of seq.

target : the target value of the function

Returns: **value** : the input value that yields the target solution. If there is no exact solution in the input sequence, finds the nearest value k such that function(k) <= target < function(k+1). This is similar to the behavior of bisect_left in the bisect package. If even the first, leftmost value of seq does not satisfy this condition, -1 is returned.

`powerlaw.ccdf`(*data*, *survival=True*, ***kwargs*)[source]¶

The complementary cumulative distribution function (CCDF) of the data.

Parameters: **data** : list or array, optional

survival : bool, optional

Whether to calculate a CDF (False) or CCDF (True). True by default.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`powerlaw.cdf`(*data*, *survival=False*, ***kwargs*)[source]¶

The cumulative distribution function (CDF) of the data.

Parameters: **data** : list or array, optional

survival : bool, optional

Whether to calculate a CDF (False) or CCDF (True). False by default.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`powerlaw.checkunique`(*data*)[source]¶

Quickly checks if a sorted array is all unique elements.

`powerlaw.cumulative_distribution_function`(*data*, *xmin=None*, *xmax=None*, *survival=False*, ***kwargs*)[source]¶

The cumulative distribution function (CDF) of the data.

Parameters: **data** : list or array, optional

survival : bool, optional

Whether to calculate a CDF (False) or CCDF (True). False by default.

xmin : int or float, optional

The minimum data size to include. Values less than xmin are excluded.

xmax : int or float, optional

The maximum data size to include. Values greater than xmin are excluded.

Returns: **X** : array

The sorted, unique values in the data.

probabilities : array

The portion of the data that is less than or equal to X.

`powerlaw.is_discrete`(*data*)[source]¶

Checks if every element of the array is an integer.

`powerlaw.loglikelihood_ratio`(*loglikelihoods1*, *loglikelihoods2*, *nested=False*, *normalized_ratio=False*)[source]¶

Calculates a loglikelihood ratio and the p-value for testing which of two probability distributions is more likely to have created a set of observations.

Parameters: **loglikelihoods1** : list or array

The logarithms of the likelihoods of each observation, calculated from a particular probability distribution.

loglikelihoods2 : list or array

The logarithms of the likelihoods of each observation, calculated from a particular probability distribution.

nested : bool, optional

Whether one of the two probability distributions that generated the likelihoods is a nested version of the other. False by default.

normalized_ratio : bool, optional

Whether to return the loglikelihood ratio, R, or the normalized ratio R/sqrt(n*variance)

Returns: **R** : float

The loglikelihood ratio of the two sets of likelihoods. If positive, the first

set of likelihoods is more likely (and so the probability distribution that produced them is a better fit to the data). If negative, the reverse is true.

p : float

The significance of the sign of R. If below a critical value (typically .05) the sign of R is taken to be significant. If above the critical value the sign of R is taken to be due to statistical fluctuations.

`powerlaw.nested_loglikelihood_ratio`(*loglikelihoods1*, *loglikelihoods2*, ***kwargs*)[source]¶

Calculates a loglikelihood ratio and the p-value for testing which of two probability distributions is more likely to have created a set of observations. Assumes one of the probability distributions is a nested version of the other.

Parameters: **loglikelihoods1** : list or array

The logarithms of the likelihoods of each observation, calculated from a particular probability distribution.

loglikelihoods2 : list or array

The logarithms of the likelihoods of each observation, calculated from a particular probability distribution.

nested : bool, optional

Whether one of the two probability distributions that generated the likelihoods is a nested version of the other. True by default.

normalized_ratio : bool, optional

Whether to return the loglikelihood ratio, R, or the normalized ratio R/sqrt(n*variance)

Returns: **R** : float

The loglikelihood ratio of the two sets of likelihoods. If positive, the first set of likelihoods is more likely (and so the probability distribution that produced them is a better fit to the data). If negative, the reverse is true.

p : float

The significance of the sign of R. If below a critical value (typically .05) the sign of R is taken to be significant. If above the critical value the sign of R is taken to be due to statistical fluctuations.

`powerlaw.pdf`(*data*, *xmin=None*, *xmax=None*, *linear_bins=False*, ***kwargs*)[source]¶

Returns the probability density function (normalized histogram) of the data.

Parameters: **data** : list or array

xmin : float, optional

Minimum value of the PDF. If None, uses the smallest value in the data.

xmax : float, optional

Maximum value of the PDF. If None, uses the largest value in the data.

linear_bins : float, optional

Whether to use linearly spaced bins, as opposed to logarithmically spaced bins (recommended for log-log plots).

Returns: **bin_edges** : array

The edges of the bins of the probability density function.

probabilities : array

The portion of the data that is within the bin. Length 1 less than bin_edges, as it corresponds to the spaces between them.

`powerlaw.plot_cdf`(*data*, *ax=None*, *survival=False*, ***kwargs*)[source]¶

Plots the cumulative distribution function (CDF) of the data to a new figure or to axis ax if provided.

Parameters: **data** : list or array

ax : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

survival : bool, optional

Whether to plot a CDF (False) or CCDF (True). False by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

`powerlaw.plot_pdf`(*data*, *ax=None*, *linear_bins=False*, ***kwargs*)[source]¶

Plots the probability density function (PDF) to a new figure or to axis ax if provided.

Parameters: **data** : list or array

ax : matplotlib axis, optional

The axis to which to plot. If None, a new figure is created.

linear_bins : bool, optional

Whether to use linearly spaced bins (True) or logarithmically spaced bins (False). False by default.

Returns: **ax** : matplotlib axis

The axis to which the plot was made.

powerlaw.trim_to_range(*data*, *xmin=None*, *xmax=None*, ***kwargs*)[source]¶

Removes elements of the data that are above xmin or below xmax (if present)

Indices and tables¶

- Index
- Module Index
- Search Page

Table Of Contents

Related Topics

- Documentation overview

This Page

- Show Source

Quick search

Go