lmoments3 Library Documentation

Release 1.0.5+0.gf231177.dirty

Ouranos Inc., Trevor James Smith, Pascal Bourgault, Florenz A. P

Jun 23, 2023

TABLE OF CONTENTS:

Version 1.0

> "In statistics, L-moments are a sequence of statistics used to summarize the shape of a probability distribution. They are L-statistics (linear combinations of order statistics, hence the "L" for "linear") analogous to conventional moments, and can be used to calculate quantities analogous to standard deviation, skewness and kurtosis, termed the L-scale, L-skewness and L-kurtosis respectively (the L-mean is identical to the conventional mean). Standardised L-moments are called L-moment ratios and are analogous to standardized moments. Just as for conventional moments, a theoretical distribution has a set of population L-moments. Sample L-moments can be defined for a sample from the population, and can be used as estimators of the population L-moments."
>
> —Wikipedia

The `lmoments3` library can be used to:

1. Calculate L-moments from sample data.
2. Fit probability distributions to sample data using L-moments.

The library extends the scipy.stats module with additional methods.

Tip: Distributions supported by *lmoments3* are documented in *Probability distributions*.

CHAPTER

ONE

LMOMENTS3

A Python 3.x library to estimate linear moments for statistical distribution functions.

Requires the packages *numpy* and *scipy*.

For the Python 2.x compatible package see *lmoments* on the Python Package Index.

1.1 Installation

lmoments3 can be installed via *pip*

```
$ pip install lmoments3
```

1.2 Documentation

Documentation is available on Read the Docs.

Source code can be found at GitHub.

1.3 Origin

This package contains a Python 3.x implementation of the lmoments.f library created by J. R. M. Hosking. (copy of original code)

1.3.1 IBM software disclaimer

The base Fortran code is copyright of the IBM Corporation, and the licensing information is shown below:

LMOMENTS: Fortran routines for use with the method of L-moments

Permission to use, copy, modify and distribute this software for any purpose and without fee is hereby granted, provided that this copyright and permission notice appear on all copies of the software. The name of the IBM Corporation may not be used in any advertising or publicity pertaining to the use of the software. IBM makes no warranty or representations about the suitability of the software for any purpose. It is provided "AS IS" without any express or implied warranty, including the implied warranties of merchantability, fitness for a particular purpose and non-infringement. IBM shall not be liable for any direct, indirect, special or consequential damages resulting from the loss of use, data or projects, whether in an action of contract or tort, arising out of or in connection with the use or performance of this software.

1.3.2 Additional code

Additional code from the R library *lmomco* has been converted into Python. This library was developed by William Asquith, and was released under the GPL-3 License. Copyright (C) 2012 William Asquith.

1.3.3 Original Python translation

The Python translation was conducted by:

Sam Gillespie
Numerical Analyst
C&R Consulting
Townsville Australia
September 2013

For more information, or to report bugs, contact: <sam.gillespie@my.jcu.edu.au>

1.3.4 Python 3 compatibility

The Python package was further updated to make it compatible with Python 3.x by Florenz A.P. Hollebrandse <f.a.p.hollebrandse@protonmail.ch>.

The software remains licenced under the GNU General Public License, see <https://www.gnu.org/licenses/gpl.html>.

The Python 3 port, is based on the original lmoments package, version 0.2.2.

The Ouranosinc package was forked from OpenHydrology's lmoments3. The primary aims of this fork are to provide maintenance for the existing codebase as well as update the project to benefit from more modern Python coding conventions, maintain compatibility with existing Python dependencies.

CHAPTER

TWO

PROBABILITY DISTRIBUTIONS

The probability distributions supported by `lmoments3` are summarised in the table below.

Name	*lmoments3* name	*scipy* name	Parameters
Exponential	*exp*	*expon*	*loc*, *scale*
Gamma	*gam*	*gamma*	*a*, *loc*, *scale* (The location parameter is not calculated using L-moments and assumed to be zero.)
Generalised Extreme Value	*gev*	*genextreme*	*c*, *loc*, *scale*
Generalised Logistic	*glo*	n/a	*k*, *loc*, *scale*
Generalised Normal	*gno*	n/a	*k*, *loc*, *scale*
Generalised Pareto	*gpa*	*genpareto*	*c*, *loc*, *scale*
Gumbel	*gum*	*gumbel_r*	*loc*, *scale*
Kappa	*kap*	n/a	*k*, *h*, *loc*, *scale*
Normal	*nor*	*norm*	*loc*, *scale*
Pearson III	*pe3*	*pearson3*	*skew*, *loc*, *scale*
Wakeby	*wak*	n/a	*beta*, *gamma*, *delta*, *loc*, *scale*
Weibull	*wei*	*weibull_m*	*c*, *loc*, *scale*

All distributions in the table above are included in the *`lmoments3.distr`* module. They can be used like any other `scipy.stats.rv_continous` distribution (documentation).

Basic usage is as follows:

```
from lmoments3 import distr

general_exp_distr = distr.exp
```

Or a distribution with parameters "frozen":

```
exp_distr = distr.exp(loc=0, scale=2)
```

CHAPTER

THREE

USING LMOMENTS3

3.1 L-moments estimation from sample data

The primary purpose of this library is to estimate L-moments from a sample dataset.

The function `lmoments3.lmom_ratios(data, nmom)()` takes an input list or *numpy* array *data* and the number of L-moments to estimate from the dataset.

```
import lmoments3 as lm

data = [2.0, 3.0, 4.0, 2.4, 5.5, 1.2, 5.4, 2.2, 7.1, 1.3, 1.5]
lm.lmom_ratios(data, nmom=5)
[Out]: [
    3.2363636363636363,
    1.1418181818181818,
    0.27388535031847133,
    0.023354564755838598,
    -0.042462845010615709,
]
```

This returns the first five sample L-moments, in the structured as l1, l2, t3, t4, t5. Where t3..5 = l3..5 / l2.

3.2 Fitting distribution functions to sample data

Sample data can be fitted directly to statistical distribution functions using the `lmoments3` library.

For example, using the gamma distribution:

```
import lmoments3 as lm
from lmoments3 import distr

data = [2.0, 3.0, 4.0, 2.4, 5.5, 1.2, 5.4, 2.2, 7.1, 1.3, 1.5]
paras = distr.gam.lmom_fit(data)
paras
[Out]: OrderedDict(
    [("a", 2.295206110128833), ("loc", 0), ("scale", 1.4100535991436407)]
)
```

This returns the distribution's parameters as an `OrderedDict` in the same order as a standard *scipy* list of distribution function parameters. The distribution parameters can be used, for example, like this:

```
fitted_gam = distr.gam(**paras)
median = fitted_gam.ppf(0.5)
median
[Out]: 2.7804212925067344
```

For full details of distribution function methods, see the scipy.stats documentation. Some useful methods include:

- *pdf*: Probability density function
- *cdf*: Cumulative distribution function
- *ppf*: Inverse cumulative distribution function (also known as quantile function or percentage point function)
- *rvs*: Random numbers generator

3.3 Computing L-moments from distribution parameters

The `lmoments3` package provides two additional methods to compute the L-moments (1..n) or L-moment ratios (1, 2, 3..n) for a distribution with given parameters.

Example:

```
distr.gam.lmom(nmom=3, **paras)
[Out]: [3.2363636363636363, 1.1418181181569327, 0.24963415541016151]

distr.gam.lmom_ratios(nmom=4, **paras)
[Out]: [
    3.2363636363636363,
    1.1418181181569327,
    0.21862865148182167,
    0.13877337951549581,
]
```

Or using the frozen distribution:

```
moments = fitted_gam.lmom(nmom=3)
ratios = fitted_gam.lmom_ratios(nmom=4)
```

3.4 Modified implementation of negative log likelihood function

`nnlf(data, *args, **kwds)()`

Calculates the Negative Log Likelihood. Provide data to calculate the negeative log likelihood. If no distribution parameters are provided, the *scipy* defaults of *loc=0* and *scale=1* are used.

Example: Calculate the Negative Log Likelihood of a Gamma distribution fitted to *data*:

```
from lmoments3 import distr

paras = distr.gam.lmom_fit(data)
distr.gam.nnlf(data, **paras)
[Out]: 21.283995091031549
```

Example: Calculate the Negative Log Likelihood of a Gamma distribution with parameters 2.5 and 1.0 when fitted to *data*:

```
from lmoments3 import distr
from collections import OrderedDict

distr.gam.nnlf(data, a=2.5, scale=1)
[Out]: 22.166452544264637
```

3.5 Other statistical methods

The *lmoments3.stats* module provides some additional statistical parametes to evaluate fitting of data to distribution function.

`AIC(data, distr_name, distr_paras)()`

Calculate the Akaike Information Criterion (AIC) using the chosen dataset and distribution.

Example: Calculate the Akaike Information Criterion for the weibull distribution using the input dataset *data*:

```
from lmoments3 import stats, distr

paras = {"loc": 0.67, "scale": 2.71, "c": 1.18}
stats.AIC(data, "wei", paras)
[Out]: 47.500528639652515
```

Functions `AICc()` and `BIC()` have a similar structure and calculate the corrected Akaike Information Criterion and the Bayesian Information Criterion respectively.

CHAPTER

FOUR

CHANGELOG

4.1 version 1.0.5 (2023-02-28)

- Rename *frechet_r_gen* to *weibull_min_r*, *frechet_r* being deprecated with SciPy 1.6
- **Migrated organisations from OpenHydrology to Ouranosinc:**
  - Added Continuous Integration checks, pre-commit configurations, package metadata adjustments
  - Code style now follows Black v2023.1
  - ReadTheDocs documentation for lmoments3 no longer nested within OpenHydrology
  - Documentation is now structured across pages
- Updated versioneer to v0.28

4.2 version 1.0.3 (2015-09-24)

- Test on Python 3.5
- Simplify doc config

4.3 version 1.0.2 (2015-02-11)

- Conda package build fix

4.4 version 1.0.1 (2014-12-02)

- Fix *lmom_fit* for *gev* distribution. Did not return *OrderedDict* in some cases.

4.5 version 1.0.0 (2014-12-01)

- Major rewrite with all distributions now being standard scipy *rv_continous* classes
- *samlmu* renamed to *lmom_ratios*
- Replaced many functions by standard scipy implementations
- General bugfixes, code improvements and test

4.6 version 0.3.1 (2014-09-25)

- Fixed setup
- Refactored unit tests

4.7 version 0.3.0 (2014-09-24)

- First release for Python 3.x

4.8 version 0.2.2 (2014-09-23)

- Improved samlmu function to support any value of nmom
- Added Bayesian Information Criterion (BIC) function
- Fixed import glitch that allowed users to import container files
- General Bugfixes

4.9 version 0.2.1 (2014-01-15)

- Support for lists as F inputs for all QUA functions
- Added Random Number Generator (rand) for all functions
- Split the main lmoments.py file into several files, as the project is getting to large to maintain as one single file.

4.10 version 0.2.0 (2014-01-13)

- Added Probability Density Functions (PDF)
- Added Reverse Lmoment Estimation Functions (LMOM)
- Added Negative Log-Likelihood Function (NlogL)
- Included Unit Tests
- Implemented better version of PELWAK function
- Support for lists as x inputs for all CDF functions

- Bugfixes
- Now licensed under the GPLv3

4.11 version 0.1.1

- Corrected Small Errors and Typos

4.12 version 0.1.0

- Initial Release

CHAPTER

FIVE

LMOMENTS3 API

5.1 Sample L-moments

lmoments3.**lmom_ratios**(*data*, *nmom=5*)

Estimate *nmom* number of L-moments from a sample *data*.

Parameters

- **data** (*list or array-like sequence*) – Sequence of (sample) data
- **nmom** (*int*) – number of L-moments to estimate

Returns

L-moment ratios like this: l1, l2, t3, t4, t5, .. . As in: items 3 and higher are L-moment ratios. # noqa: E501

Return type

list

5.2 Distributions

class lmoments3.distr.**LmomDistrMixin**

Mixin class to add L-moment methods to `scipy.stats.rv_continous` distribution functions. Distributions using the mixin should override the methods `_lmom_fit()` and *lmom_ratios()*.

lmom_fit(*data=[]*, *lmom_ratios=[]*)

Fit the distribution function to the given data or given L-moments.

Parameters

- **data** (*array_like*) – Data to use in calculating the distribution parameters
- **lmom_ratios** (*array_like*) – L-moments (ratios) l1, l2, t3, t4, .. to use in calculating the distribution parameters

Returns

Distribution parameters in *scipy* order, e.g. scale, loc, shape

Return type

`OrderedDict`

lmom(**args*, *nmom=5*, ***kwds*)

Compute the distribution's L-moments, e.g. l1, l2, l3, l4, ..

Parameters

- **args** (*float*) – Distribution parameters in order of shape(s), loc, scale
- **nmom** (*int*) – Number of moments to calculate
- **kwds** (*float*) – Distribution parameters as named arguments. See `rv_continous.shapes` for names of shape parameters

Returns
List of L-moments

Return type
list

lmom_ratios(**args*, *nmom=5*, ***kwds*)

Compute the distribution's L-moment ratios, e.g. l1, l2, t3, t4, ..

Parameters

- **args** (*float*) – Distribution parameters in order of shape(s), loc, scale
- **nmom** (*int*) – Number of moments to calculate
- **kwds** (*float*) – Distribution parameters as named arguments. See `rv_continous.shapes` for names of shape parameters

Returns
List of L-moment ratios

Return type
list

nnlf(*data*, **args*, ***kwds*)

5.3 Other statistical functions

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

L

M

N