pyscenarios Documentation

Release 0.2.1

pyscenarios Developers

2019-05-01

CONTENTS

1 Index **3**
- 1.1 Installation . . . 3
- 1.2 What's New . . . 3
- 1.3 copula . . . 4
- 1.4 sobol . . . 6
- 1.5 stats . . . 6

2 License **9**

Python Module Index **11**

This package offers several modules used in Monte Carlo simulations:

copula Gaussian Copula, Student T* Copula and IT Copula samples generators

sobol Joe/Kuo Sobol series generator

stats Tail dependence measures

All modules fully support and are optimized for dask and are compatible with dask distributed.

CHAPTER

ONE

INDEX

1.1 Installation

1.1.1 Required dependencies

- Python 3.5 or later
- dask
- numba
- numpy
- scipy

1.1.2 Testing

To run the test suite after installing pyscenarios, first install (via pypi or conda)

- py.test: Simple unit testing library

and run `py.test`.

1.2 What's New

1.2.1 v0.2.1 (2019-05-01)

- Make package discoverable by mypy
- A more robust fix for dask#4739

1.2.2 v0.2.0 (2019-04-29)

- Type annotations
- 'rng' parameter in copula functions is now case insensitive
- Work around regression in IT copula with dask >= 1.1 (dask#4739)
- Smaller binary package; simplified setup
- Explicit CI tests for Windows, Python 3.5.0, and Python 3.7

- Mandatory flake8 and mypy in CI
- Changed license to Apache 2.0

1.2.3 v0.1.0 (2018-05-27)

Initial release.

1.3 copula

High performance copula generators

pyscenarios.copula.**gaussian_copula**(*cov: Union[List[List[float]], numpy.ndarray], samples: int, seed: int = 0, chunks: Union[None, int, Tuple[int, int], Tuple[Tuple[int, ...], Tuple[int, ...]]] = None, rng: str = 'Mersenne Twister'*) → Union[numpy.ndarray, dask.array.core.Array]

Gaussian Copula scenario generator.

Simplified algorithm:

```
>>> l = numpy.linalg.cholesky(cov)
>>> y = numpy.random.standard_normal(size=(samples, cov.shape[0]))
>>> p = (l @ y.T).T
```

Parameters

- **cov** (*numpy.ndarray*) – covariance matrix, a.k.a. correlation matrix. It must be a Hermitian, positive-definite matrix in any square array-like format. The width of cov determines the number of dimensions of the output.
- **samples** (*int*) – Number of random samples to generate

  Note: When using Sobol, to obtain a uniform distribution one must use $2^n - 1$ samples (for any n > 0).

- **chunks** – Chunk size for the return array, which has shape (samples, dimensions). It can be anything accepted by dask (a positive integer, a tuple of two ints, or a tuple of two tuples of ints) for the output shape.

  Set to None to return a numpy array.

  Warning: When using the Mersenne Twister random generator, the chunk size changes the random sequence. To guarantee repeatability, it must be fixed together with the seed. chunks=None also produces different results from using dask.

- **seed** (*int*) – Random seed.

  With `rng='Sobol'`, this is the initial dimension; when generating multiple copulas with different seeds, one should never use seeds that are less than `cov.shape[0]` apart from each other.

  The maximum seed when using sobol is:

```
pyscenarios.sobol.max_dimensions() - cov.shape[0] - 1
```

- **rng** (*str*) – Either `Mersenne Twister` or `Sobol`

Returns array of shape (samples, dimensions), with all series being normal (0, 1) distributions.

Return type If chunks is not None, `dask.array.Array`; else `numpy.ndarray`

`pyscenarios.copula.`**`t_copula`**(*cov: Union[List[List[float]], numpy.ndarray], df: Union[int, List[int], numpy.ndarray], samples: int, seed: int = 0, chunks: Union[None, int, Tuple[int, int], Tuple[Tuple[int, ...], Tuple[int, ...]]] = None, rng: str = 'Mersenne Twister'*) → Union[numpy.ndarray, dask.array.core.Array]

Student T Copula / IT Copula scenario generator.

Simplified algorithm:

```
>>> l = numpy.linalg.cholesky(cov)
>>> y = numpy.random.standard_normal(size=(samples, cov.shape[0]))
>>> p = (l @ y.T).T  # Gaussian Copula
>>> r = numpy.random.uniform(size=(samples, 1))
>>> s = scipy.stats.chi2.ppf(r, df=df)
>>> z = numpy.sqrt(df / s) * p
>>> u = scipy.stats.t.cdf(z, df=df)
>>> t = scipy.stats.norm.ppf(u)
```

Parameters

- **cov** (*numpy.ndarray*) – covariance matrix, a.k.a. correlation matrix. It must be a Hermitian, positive-definite matrix in any square array-like format. The width of cov determines the number of dimensions of the output.
- **df** – Number of degrees of freedom. Can be either a scalar int for Student T Copula, or a one-dimensional array-like with one point per dimension for IT Copula.
- **samples** (*int*) – Number of random samples to generate

  Note: When using Sobol, to obtain a uniform distribution one must use $2^n - 1$ samples (for any n > 0).

- **chunks** – Chunk size for the return array, which has shape (samples, dimensions). It can be anything accepted by dask (a positive integer, a tuple of two ints, or a tuple of two tuples of ints) for the output shape.

  Set to None to return a numpy array.

  Warning: When using the Mersenne Twister random generator, the chunk size changes the random sequence. To guarantee repeatability, it must be fixed together with the seed. chunks=None also produces different results from using dask.

- **seed** (*int*) – Random seed.

  With `rng='Sobol'`, this is the initial dimension; when generating multiple copulas with different seeds, one should never use seeds that are less than `cov.shape[0] + 1` apart from each other.

The maximum seed when using sobol is:

```
pyscenarios.sobol.max_dimensions() - cov.shape[0] - 2
```

- **rng** (*str*) – Either `Mersenne Twister` or `Sobol`

Returns array of shape (samples, dimensions), with all series being normal (0, 1) distributions.

Return type If chunks is not None, `dask.array.Array`; else `numpy.ndarray`

1.4 sobol

Sobol sequence generator

This is a reimplementation of a C++ algorithm by Stephen Joe and Frances Y. Kuo. Directions are based on `new-joe-kuo-6.21201` from the URL above.

`pyscenarios.sobol.`**`sobol`**(*size: Union[int, Tuple[int, int]], d0: int = 0, chunks: Union[None, int, Tuple[int, int], Tuple[Tuple[int, ...], Tuple[int, ...]]] = None*) → Union[numpy.ndarray, dask.array.core.Array]

Sobol points generator based on Gray code order

Parameters

- **size** – number of samples (cannot be greater than 2^{32}) to extract from a single dimension, or tuple (samples, dimensions). To guarantee uniform distribution, the number of samples should always be $2^n - 1$.
- **d0** (*int*) – first dimension. This can be used as a functional equivalent of a a random seed. dimensions + d0 can't be greater than *max_dimensions()* - 1.
- **chunks** – If None, return a numpy array.

  If set, return a dask array with the given chunk size. It can be anything accepted by dask (a positive integer, a tuple of two ints, or a tuple of two tuples of ints) for the output shape (see result below). e.g. either `(16384, 50)` or `((16384, 16383), (50, 50, 50))` could be used together with `size=(32767, 150)`.

  Note: The algorithm is not efficient if there are multiple chunks on axis 0. However, if you do need them, it is typically better to require them here than re-chunking afterwards, particularly if (most of) the subsequent algorithm is embarassingly parallel.

Returns If size is an int, a 1-dimensional array of samples. If size is a tuple, a 2-dimensional array POINTS, where `POINTS[i, j]` is the ith sample of the jth dimension. Each dimension is a uniform (0, 1) distribution.

Return type If chunks is not None, `dask.array.Array`; else `numpy.ndarray`

`pyscenarios.sobol.`**`max_dimensions`**() → int

Return number of dimensions available. When invoking *sobol()*, `size[1] + d0` must be smaller than this.

1.5 stats

Statistical functions

pyscenarios.stats.**tail_dependence**(*x: Any*, *y: Any*, *q: Any*) → Union[numpy.ndarray, dask.array.core.Array]

Calculate tail dependence between vectors x and y.

Parameters

- **x** – 1D array-like or dask array containing samples from a uniform (0, 1) distribution.
- **y** – other array to compare against
- **q** – quantile(s) (0 < q < 1). Either a scalar or a ND array-like or dask array.

Returns

array of the same shape and type as q, containing:

$$P(y < q|x < q)|q < 0.5$$
$$P(y \geq q|x \geq q)|q \geq 0.5$$

CHAPTER

TWO

LICENSE

pyscenarios is available under the open source Apache License.

PYTHON MODULE INDEX

p

pyscenarios.copula, 4
pyscenarios.sobol, 6
pyscenarios.stats, 6

INDEX

G

gaussian_copula() (*in module pyscenarios.copula*), 4

M

max_dimensions() (*in module pyscenarios.sobol*), 6

P

pyscenarios.copula (*module*), 4
pyscenarios.sobol (*module*), 6
pyscenarios.stats (*module*), 6

S

sobol() (*in module pyscenarios.sobol*), 6

T

t_copula() (*in module pyscenarios.copula*), 5
tail_dependence() (*in module pyscenarios.stats*), 6