SciPy 2025
July 7 - July 13, 2025

Proceedings of the 24th Python in Science Conference
ISSN: 2575-9752

**Wael Tabbara**[1], **Sharafeddine Sharafeddine**[2,1], **Ahmad El-Hajj**[3], **Jihad Fahs**[1], and **Ibrahim Abou-Faycal**[1]

[1]Electrical & Computer Eng. Department, American University of Beirut, Lebanon., [2]School of Electrical & Computer Engineering, Cornell University, USA., [3]Electrical & Computer Eng. Department, American University of Beirut, Lebanon

## Abstract

Heavy-tailed distributions are increasingly found to better fit empirical data in engineering, finance, physics, network science, and related fields. Among them, $\alpha$-stable distributions play a central role being limiting laws in the generalized central limit theorem: they are expected to be exceptionally good models whenever sums of multiple independent heavy-tailed sources are at play. Despite their theoretical importance, their practical use remains challenging: $\alpha$ -stable probability densities generally do not have closed-form expressions, and numerical evaluation and random variate generation can be difficult, especially in the multivariate setting. This paper presents AUB-HTP, a Python package for numerical computation and simulation of $\alpha$-stable distributions. The package provides scalar density evaluation using several complementary methods, including Zolotarev-type integral representations, series formulas, and numerical inversion of characteristic functions. It also provides random variate generation for scalar and multivariate $\alpha$-stable distributions, with support for flexible spectral measures through LePage series representations. Numerical experiments demonstrate that AUB-HTP improves the accuracy, stability, and parameter coverage of existing tools for scalar density computation, while adding new capabilities for multivariate simulation. The package is designed to support reproducible computational work involving heavy-tailed models across a broad range of scientific applications.

**Keywords** Heavy Tailed Data, Heavy Tails, Alpha Stable, Levy Stable

## 1. Introduction

Heavy-tailed distributions arise naturally in many scientific and engineering applications, including communications, signal processing, finance, physics, and network modeling M. Nassar, K. Gulati, A. Sujeeth, N. Aghasadeghi, B. Evans, and K. Tinsley [1], G. Tsihrintzis, P. Tsakalides, and C. Nikias [2], M. E. Crovella, M. S. Taqqu, and A. Bestavros [3], J. P. Nolan [4], N. N. Taleb [5]. In such settings, rare but large deviations occur more frequently than predicted by light-tailed models such as the Gaussian distribution. As a result, Gaussian-based modeling can significantly underestimate the probability and impact of extreme events.

Among heavy-tailed models, $\alpha$-stable distributions occupy a central theoretical position. They generalize the Gaussian distribution through the stability property: sums of independent $\alpha$-stable random variables remain $\alpha$-stable, up to location and scale changes. They also arise as limiting distributions for normalized sums of independent random variables under the generalized central limit theorem B. V. Gnedenko and A. N. Kolmogorov [6]. These properties make $\alpha$-stable laws natural candidates for modeling aggregate effects generated by many independent heavy-tailed sources.

**Published** Jul 13, 2026

**Correspondence to**
Ahmad El-Hajj
aae108@mail.aub.edu

A major practical limitation of $\alpha$-stable distributions is that their probability density functions (PDF)s generally do not admit closed-form expressions, except in the special cases of the Cauchy and Lévy distributions. Consequently, numerical tools are required for density evaluation, modeling, and simulation. Existing approaches for scalar $\alpha$-stable random variate generation, such as the Chambers–Mallows–Stuck (CMS) method J. M. Chambers, C. L. Mallows, and B. W. Stuck [7], are well established, but they do not directly address general multivariate $\alpha$-stable distributions. In the multivariate case, available simulation methods typically focus on special spectral measures, such as isotropic or sub-Gaussian models J. P. Nolan [4], rather than arbitrary spectral measures.

This paper presents AUB-HTP, a Python package for computational work with $\alpha$-stable distributions. The package provides scalar density computation using several numerical methods, including Zolotarev integral representations V. M. Zolotarev [8], Skorohod-Pollard-Bergström formulas A. V. Skorohod [9], H. Pollard [10], H. Bergström [11], and direct numerical inversion of the *characteristic function*[1]. These methods provide complementary tradeoffs in terms of accuracy, speed, and robustness across parameter regimes. In addition, AUB-HTP implements random variate generation for scalar and multivariate $\alpha$ -stable distributions. For the multivariate case, the package uses LePage series representations R. LePage, M. Woodroofe, and J. Zinn [12], which support flexible spectral measures and perform particularly well in challenging heavy-tailed regimes, including small values of $\alpha$.

The goal of AUB-HTP is to make $\alpha$-stable modeling more accessible to computational scientists by providing reliable density evaluation, flexible simulation tools, and reproducible numerical routines within the Python ecosystem. The main contributions of this work are: (i) a unified Python implementation of several scalar $\alpha$-stable PDF evaluation methods; (ii) a flexible random variate generator for scalar and multivariate $\alpha$-stable distributions; (iii) support for general spectral measures through LePage series; and (iv) numerical benchmarks comparing accuracy, runtime, and robustness against existing implementations.

### *Notations*

We adopt the following conventions. The vector $\boldsymbol{x} \in \mathbb{R}^d$ represents a column vector and $\boldsymbol{x}^T$ is its transpose. The Euclidean inner product of two vectors $\boldsymbol{x} = (x_1, x_2, \cdots, x_d)^T$ and $\boldsymbol{y} = (y_1, y_2, \cdots, y_d)^T$ is denoted by $\boldsymbol{x}^T\boldsymbol{y} = \langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{d} x_i y_i$ and the $L_2$ norm of $\boldsymbol{x}$ by $\mid \boldsymbol{x} \mid = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle}$. The unit sphere in $\mathbb{R}^d$ is denoted by $\mathbb{S}^{d-1} = \{\boldsymbol{x} \in \mathbb{R}^d : \mid \boldsymbol{x} \mid = 1\}$. The class of all borel subsets in a metric space $S$ is represented by $\mathcal{B}(S)$. We use $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y}$ to indicate that the two random vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$ are independent. When a random vector $\boldsymbol{X}$ is distributed uniformly on the unit sphere, we write $\boldsymbol{X} \sim \mathcal{U}(\mathbb{S}^{d-1})$. For a random variable $X$ that is Gamma distributed, we write $X \sim \mathrm{Gamma}(\alpha, \beta)$, and we use the following parameterization for the PDF

$$f_X(x) = \begin{cases} \frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-\frac{x}{\beta}} & \text{if } x > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\Gamma(\cdot)$ is the Gamma function.

## 2. Preliminaries on $\alpha$-Stable Distributions

We adopt the definition of a multivariate stable vector given in G. Samorodnitsky and M. S. Taqqu [13]:

[1] The characteristic function of a random variable $X$ is given by the expected value of $e^{\mathrm{i}tX}$, $\phi_X(u) = \mathbb{E}[e^{\mathrm{i}uX}]$. It uniquely determines the distribution of $X$, and it always exists.

**Definition 1.**

A random vector $\boldsymbol{X} = (X_1, X_2, \cdots, X_d)^T$ is said to be a stable random vector in $\mathbb{R}^d$ if for any positive numbers A and B there is a positive number C and a vector $\boldsymbol{D} \in \mathbb{R}^d$ such that

$$A\boldsymbol{X}^{(1)} + B\boldsymbol{X}^{(2)} = C\boldsymbol{X} + \boldsymbol{D}, \tag{2}$$

where $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(2)}$ are independent copies of $\boldsymbol{X}$. The vector $\boldsymbol{X}$ is called strictly stable if (1) holds with $\boldsymbol{D} = \boldsymbol{0}$ for any $A > 0$ and $B > 0$.

**Definition 2.**

G. Samorodnitsky and M. S. Taqqu [13]

Let $0 < \alpha < 2$, then $\boldsymbol{X} = (X_1, X_2, \cdots, X_d)$ is an $\alpha$-stable random vector in $\mathbb{R}^d$ if and only if there exists a finite measure $\Lambda$ on $\mathbb{S}^{d-1}$ called the "spectral" measure and a vector $\boldsymbol{\mu}^0$ in $\mathbb{R}^d$ such that the characteristic function of $\boldsymbol{X}$ is given by

$$\phi_{\boldsymbol{X}}(\boldsymbol{t}) = \mathbb{E}[\exp\{\mathrm{i}\langle \boldsymbol{t}, \boldsymbol{X}\rangle\}] = \exp\left\{-\int_{\mathbb{S}^{d-1}} \psi_\alpha(\langle \boldsymbol{t}, \boldsymbol{s}\rangle)\, \Lambda(d\boldsymbol{s}) + \mathrm{i}\langle \boldsymbol{t}, \boldsymbol{\mu}^0\rangle\right\}, \tag{3}$$

and where

$$\psi_\alpha(u) = \begin{cases} |u|^\alpha \left[1 - \mathrm{i}\,\mathrm{sign}(u)\tan\frac{\pi\alpha}{2}\right] & \alpha \neq 1 \\ |u| \left[1 + \mathrm{i}\frac{2}{\pi}\,\mathrm{sign}(u)\ln|u|\right] & \alpha = 1. \end{cases} \tag{4}$$

In particular, we are interested in $\alpha$-stable [2] random vectors.

In the scalar case ($d = 1$) –and whenever $0 < \alpha < 2$, the spectral measure boils down to two mass points on $s = \pm 1$, and the characteristic function in (2) is expressible in terms of 4 parameters: the stability index $0 < \alpha < 2$, skewness $-1 \leq \beta \leq 1$, scale $\gamma > 0$, and location $\delta \in \mathbb{R}$. In this case, we denote an $\alpha$-stable random variable $X$ using the same notation as in Nolan J. P. Nolan [14], $X \sim \mathcal{S}(\alpha, \beta, \gamma, \delta; k)$ where $k$ is the parameterization index (or type). We will use the symbols $\mathcal{S}_0$ and $\mathcal{S}_1$ to refer to the two common type-0 and type-1 parameterization, respectively. In the multivariate case, we denote a $d$-dimension non-degenerate $\alpha$-stable random vector $\boldsymbol{X}$ with stability index $\alpha$, spectral measure $\Lambda$ and shift vector $\boldsymbol{\mu}^0$, by $\boldsymbol{X} \sim \boldsymbol{S}_\alpha(\Lambda, d, \boldsymbol{\mu}^0)$. Note that Definition 2 corresponds to the generalization of the scalar case under the parameterization $k = 1$.

### *2.1. Parameterizations of scalar $\alpha$-stable distributions*

The two parameterizations $\mathcal{S}(\alpha, \beta, \gamma, \delta; 0)$ and $\mathcal{S}(\alpha, \beta, \gamma, \delta; 1)$ differ primarily in how they define the location (shift) of the distribution, particularly near $\alpha = 1$. For a given $\alpha$ and $\beta$, let $Z$ be a "standard" $\alpha$-stable variable corresponding to $\gamma = 1$ and $\delta = 0$.

- **Parameterization 0:**

A random variable $X \sim \mathcal{S}(\alpha, \beta, \gamma, \delta; 0)$ if

$$X \overset{d}{=} \begin{cases} \gamma(Z - \beta\tan(\frac{\pi\alpha}{2})) + \delta, & \alpha \neq 1 \\ \gamma Z + \delta, & \alpha = 1. \end{cases} \tag{5}$$

The characteristic function of $X$ is given by

[2] The term $\alpha$-stable is used to refer to the stable family excluding the Gaussian law.

$$\mathbb{E}[e^{iuX}] = \begin{cases} \exp\big(-|\gamma u|^\alpha \ [1 + \mathrm{i}\beta \tan(\frac{\pi\alpha}{2}) \operatorname{sign}(u)(|\gamma u|^{1-\alpha} - 1)] + \mathrm{i}\delta u\big), \alpha \neq 1 \\ \exp\big(-|\gamma u| \ \big[1 + \mathrm{i}\beta \frac{2}{\pi} \operatorname{sign}(u) \log|\gamma u|\big] + \mathrm{i}\delta u\big), \qquad \alpha = 1. \end{cases} \tag{6}$$

This parameterization has the advantage of avoiding discontinuities near $\alpha = 1$ by adjusting the shift behavior. It is often preferred for numerical stability J. P. Nolan [14].

- **Parameterization 1:**

A random variable $X \sim \mathcal{S}(\alpha, \beta, \gamma, \delta; 1)$ if:

$$X \overset{d}{=} \begin{cases} \gamma Z + \delta, & \alpha \neq 1 \\ \gamma Z + \big(\delta + \beta \frac{2}{\pi} \gamma \log \gamma\big), & \alpha = 1, \end{cases} \tag{7}$$

with a corresponding characteristic function given by

$$\mathbb{E}[e^{iuX}] = \begin{cases} \exp\big(-|\gamma u|^\alpha \ [1 - \mathrm{i}\beta \tan(\frac{\pi\alpha}{2}) \operatorname{sign}(u)] + \mathrm{i}\delta u\big), \alpha \neq 1 \\ \exp\big(-|\gamma u| \ \big[1 + \mathrm{i}\beta \frac{2}{\pi} \operatorname{sign}(u) \log|u|\big] + \mathrm{i}\delta u\big), \quad \alpha = 1. \end{cases} \tag{8}$$

Unlike type-0, this parameterization requires an explicit adjustment to the location when $\alpha = 1$.

Note that the choice of parameterization directly affects the shift and symmetry of the distribution, especially when $\beta \neq 0$ and $\alpha$ is close to 1. In both forms, the distribution can be standardized by setting $\gamma = 1$ and $\delta = 0$, in which case we use the abbreviations $\mathcal{S}(\alpha, \beta; 0)$ and $\mathcal{S}(\alpha, \beta; 1)$ respectively.

In the univariate case, our package can simulate both parameterizations $k = 0, 1$.

## 3. Scalar Density Evaluation

The AUB-HTP package provides numerical evaluation of scalar $\alpha$-stable probability density functions through a SciPy-like interface. This component complements the random number generator: the generator produces samples, while the density module provides the reference curve used for model verification, visualization, and numerical comparison.

Except for Cauchy and Levy laws, $\alpha$-stable distributions do not have a closed-form density, which must therefore be computed using numerical inversion or analytic representations that are valid in specific parameter regions.

Towards fulfilling our main objective of reducing expensive quadrature calls while preserving accuracy across a wide range of $\alpha$, $\beta$, and sample values, the density evaluator combines three methods:

- direct inversion of Nolan's $\mathcal{S}_1$ characteristic function near the mode J. P. Nolan [14]
- Zolotarev's integral representation in the middle range V. M. Zolotarev [8]
- Skorohod-Pollard-Bergström's series expansions in the tails A. V. Skorohod [9], H. Pollard [10], A. Wintner [15].

### *3.1. Input standardization*

All density calls are first mapped to a standardized variable allowing the internal routines to work on standardized densities while still supporting user-facing location and scale parameters. More specifically, given an input sample value $x$ the code evaluates

$$z = \frac{x - \text{shift}}{\gamma}, \tag{9}$$

then rescales the final density by $1/\gamma$. For the $\mathcal{S}_1$ parameterization for example, the shift is given by J. P. Nolan [14, equation 1.5]

$$\text{shift} = \begin{cases} \delta, & \alpha \neq 1 \\ \delta + \beta \frac{2}{\pi} \gamma \log \gamma & \alpha = 1, \end{cases} \tag{10}$$

as can be deduced from equation (5).

Additionally, if $f_{\alpha,\beta}(x)$ is the PDF of a normalized $\alpha$-stable random variable, for negative evaluation points the code uses the symmetry relation J. P. Nolan [14, proposition 1.1]

$$f_{\alpha,\beta}(-x) = f_{\alpha,-\beta}(x), \tag{11}$$

and most formulas only need to be evaluated on the positive half-line.

### *3.2. Hybrid evaluation strategy*

The main density function acts as a dispatcher. It divides the input array into regions and applies the method that is most accurate for each region.

- For small $\mid z \mid$ (see equation (7)), the package uses direct inversion of the characteristic function. While this method is slower than the series expansions, it is stable near the mode where tail formulas are not accurate.
- For intermediate values of $\mid z \mid$, the package uses Zolotarev's integral representation. This provides a useful compromise between direct inversion and asymptotic series methods.
- For large $\mid z \mid$, the package uses Skorohod-Pollard-Bergström series' expansions. These formulas are fast in the tails and avoid repeated numerical integration.

The switching points between methods are not fixed constants. They are instead selected from precomputed cutoff functions depending on $(\alpha, \beta)$. These cutoffs were obtained using grid-based comparison with `scipy.stats.levy_stable.pdf`. At runtime, the code queries the corresponding cutoff and applies Boolean masks to route each input point to the selected method.

#### 3.2.1. *Density formulas*

• For direct inversion in the $\mathcal{S}_1$ parameterization J. P. Nolan [14], the density is computed from the characteristic function. The standardized density is

$$\begin{aligned} f_{\alpha,\beta}(x) &= \frac{1}{\pi} \int_0^\infty e^{-t^\alpha} \cos\Big(xt - \beta \tan\Big(\frac{\pi\alpha}{2}\Big) t^\alpha\Big)\, dt \quad \alpha \neq 1 \\ f_{1,\beta}(x) &= \frac{1}{\pi} \int_0^\infty e^{-t} \cos\Big(xt + \frac{2}{\pi}\beta t \log t\Big)\, dt \qquad \alpha = 1, \text{ with the logarithmic correction.} \end{aligned} \tag{12}$$

These integrals are evaluated with `scipy.integrate.quad_vec`.

• For the middle region, the code uses Zolotarev's integral representation V. M. Zolotarev [8]. For $\alpha \neq 1$ and $x > 0$,

$$f_{\alpha,\beta}(x) = \frac{\alpha x^{\frac{1}{\alpha-1}}}{\pi \mid \alpha - 1 \mid} \int_{-\theta_0}^{\pi/2} V(\theta) \exp\big[-x^{\frac{\alpha}{\alpha-1}} V(\theta)\big]\, d\theta, \tag{13}$$

where

$$-\theta_0 = \frac{1}{\alpha} \arctan\Big(\beta \tan\Big(\frac{\pi\alpha}{2}\Big)\Big), \tag{14}$$

and where $V(\theta)$ is evaluated following Zolotarev's formula V. M. Zolotarev [8]. This integral is computed using the function `scipy.integrate.quad`.

• For the tail region, we use Skorohod-type series expansions A. V. Skorohod [9], H. Pollard [10], A. Wintner [15] of the form

$$f_{\alpha,\beta}(x) \approx \frac{1}{\pi x} \sum_{n=1}^{N} a_n x^{-\alpha n}, \tag{15}$$

where

$$a_n = \frac{(-1)^{n-1}\Gamma(n\alpha+1)}{n!}\Big(1+\beta^2\tan^2\Big(\frac{\pi\alpha}{2}\Big)\Big)^{n/2}\sin\Big[n\Big(-\frac{\pi\alpha}{2}+\arctan\Big(\beta\tan\Big(\frac{\pi\alpha}{2}\Big)\Big)\Big)\Big]. \tag{16}$$

The number $N$ of terms used to calculate the series is appropriately chosen depending on the parameter range. In the current package, the $\alpha < 1$ branch uses a larger truncation level, while the $\alpha > 1$ branch requires fewer terms in the tested regimes.

### *3.3. Validation*

The density evaluator was tested against `scipy.stats.levy_stable.pdf` on grids of $\alpha$, $\beta$, and $x$. The near-mode tests used $x \in [-10, 10]$, while the tail tests examined values up to $x = 100$.

The comparison recorded the following metrics:

- Maximum absolute error:

$$E_{\max}^{\text{abs}} = \max_{1\le i\le n} \mid \hat{f}(x_i) - f_{\text{SciPy}}(x_i) \mid. \tag{17}$$

- Maximum relative error:

$$E_{\max}^{\text{rel}} = \max_{1\le i\le n} \frac{\mid \hat{f}(x_i) - f_{\text{SciPy}}(x_i) \mid}{\max\big(\mid f_{\text{SciPy}}(x_i) \mid, \varepsilon\big)}, \tag{18}$$

where $\varepsilon > 0$ prevents division by zero.

- Mean absolute error:

$$E_{\text{mean}}^{\text{abs}} = \frac{1}{n}\sum_{i=1}^{n} \mid \hat{f}(x_i) - f_{\text{SciPy}}(x_i) \mid. \tag{19}$$

- Runtime against SciPy, reported through the speedup factor:

$$S = \frac{T_{\text{SciPy}}}{T_{\text{AUB -HTP}}}. \tag{20}$$

The tests show close agreement across most of the parameter space, but they also reveal regions where the SciPy reference exhibits irregular numerical behavior. The strongest agreement occurs away from the narrow region around $\alpha = 1$, where changes in the parameterization and the presence of logarithmic terms make density evaluation more sensitive. In this region, visual diagnostics are important because the relative difference may become large when the reference density is close to zero or numerically unstable. Figure 1 illustrates this behavior near $\alpha = 1$ using the maximum relative difference measured around the mode. Away from $\alpha = 1$, Figure 1 shows agreement with the SciPy reference. Results for the remaining error metrics, parameter ranges, and tail evaluations, together with the complete benchmark data, are available in the Github repository[3].

[3] https://github.com/AUB-HTP/AUB-HTP/tree/main/benchmarks

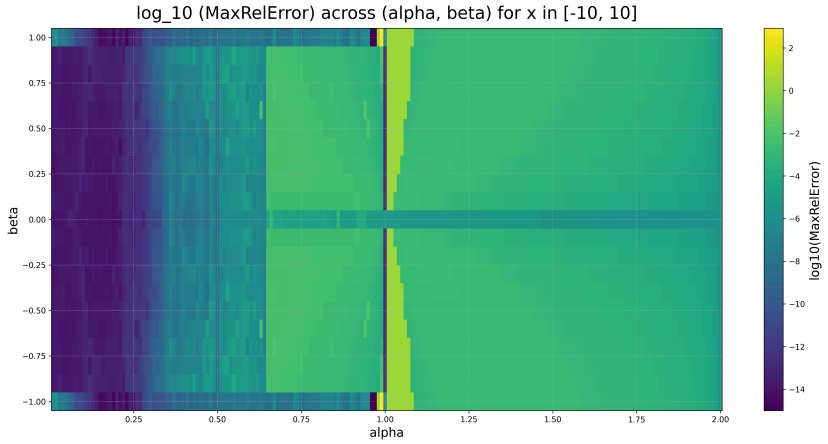



**Figure 1.** *Maximum relative difference across $(\alpha, \beta)$ near the mode. Larger values near $\alpha = 1$ mark the most sensitive numerical region.*

### *3.4. Discrepancies in the SciPy reference*

While `scipy.stats.levy_stable.pdf` The SciPy Community [16] is used as the main numerical reference, we observed that it has some discrepancies for certain parameter values. These discrepancies appear mostly near the sensitive region around $\alpha = 1$, and they can create misleading relative difference values.

The main issue is that the SciPy PDF sometimes shows unusual zero or near-zero regions in some places while the densities are known to be smooth. Since the maximum relative difference contains a division by the reference value, even a small disagreement in such regions can produce a very large relative difference. Therefore, some of the large relative differences in the benchmark tables do not necessarily mean that our PDF generator is inaccurate. Instead, they are likely due to the instability or discontinuity in the reference PDF.

To validate this, we generated visual diagnostic plots comparing the AUB-HTP with the `levy_stable` PDF values. Figure 2 shows several cases where the SciPy reference generates unexpected zero windows or sharp changes, while the PDF generated using our package remains smooth.

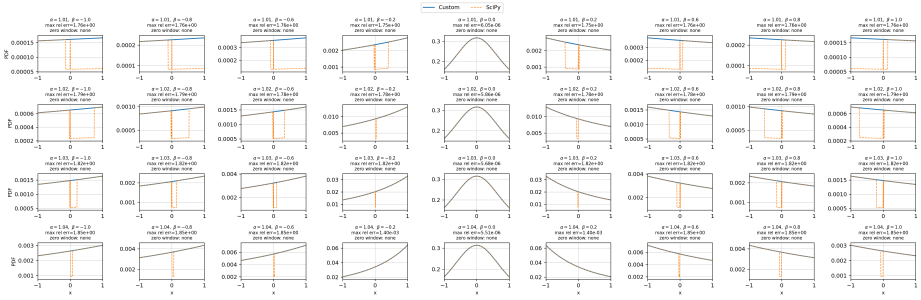


**Figure 2.** *Visual comparison between AUB-HTP PDF generator and `scipy.stats.levy_stable.pdf`. The plots zoom into regions where the SciPy reference becomes numerically zero or behaves irregularly. These regions explain why some relative difference values become large even when our generated density remains smooth.*

The heatmaps are useful for locating sensitive parameter regions, but the diagnostic plots were needed to understand whether the difference is primarily attributed to our method or the numerical reference.

### *3.5. Runtime*

The adopted hybrid strategy reduces runtime by avoiding direct quadrature when a faster formula is accurate enough. The largest gains occur in the tail region, where series expansions replace repeated numerical integration.

In the benchmark runs, the median speedup was approximately $58\times$, with larger speedups in tail-heavy evaluations. These gains come from three design choices:

- vectorized array operations
- region-specific method dispatch
- reuse of precomputed cutoff functions.

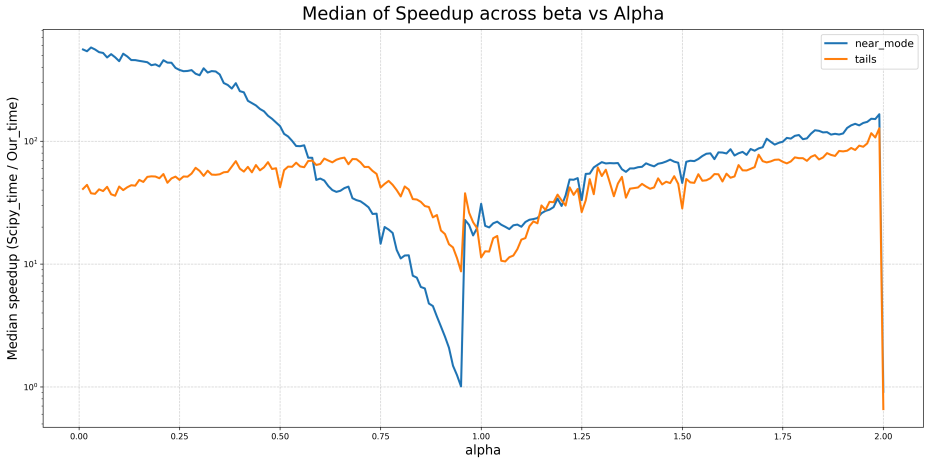



**Figure 3.** *Median runtime speedup relative to* `scipy.stats.levy_stable.pdf`*. The largest gains occur where the series expansions are selected.*

The speedup plot in Figure 3 shows that our method is strictly faster than `scipy.stats.levy_stable.pdf` for almost all tested values of $\alpha$. Note that the vertical axis is on a logarithmic scale and values between 10 and 100 represent large runtime improvements. For small values of $\alpha$, especially in the near-mode range, the speedup is very high. This is because the method selection avoids unnecessary numerical integration and uses the formulas that are faster in those regions. In the tail range, the speedup is also consistent for most values of $\alpha$, since the Skorohod-Pollard-Bergström formulas replace repeated quadrature calls. The smallest speedups occur near $\alpha = 1$ which is expected since this region is numerically delicate. The formulas change around $\alpha = 1$, and the characteristic function contains a logarithmic correction term. The package therefore relies more on direct inversion in this region, which is slower than the series-based methods.

## 4. Generating $\alpha$-Stable Random Vectors

In the univariate case, the package implements a numerically stable version of the Chambers, Mallows, and Stuck method which is presented in J. M. Chambers, C. L. Mallows, and B. W. Stuck [7], J. P. Nolan [14]. In the multivariate setup, the proposed random number generator is based on a theoretical foundation that we state next.

### *4.1. The LePage series*

The LePage series G. Samorodnitsky and M. S. Taqqu [13] was first introduced in R. LePage, M. Woodroofe, and J. Zinn [12] and allows for simulating a multivariate stable vector with a stability index $\alpha$ and a given spectral measure through the use of the following theorem V. Bentkus, A. Juozulynas, and V. Paulauskas [17, Theorem 4]

**Theorem 1**.

Let $\boldsymbol{V_1}, \boldsymbol{V_2}, \cdots$ be independent and identically distributed (i.i.d) random vectors in $\mathbb{R}^d$ that are drawn according to a spectral measure $\Lambda$ where $\Lambda\big(\mathbb{S}^{d-1}\big) > 0$, i.e.,

$$\Pr(\boldsymbol{V} \in A) = \frac{\Lambda\big(\mathbb{S}^{d-1} \cap A\big)}{\Lambda(\mathbb{S}^{d-1})} \quad \forall\, A \in \mathcal{B}\big(\mathbb{R}^d\big). \tag{21}$$

Let $E_1, E_2, \cdots$ be i.i.d. exponential random variables with mean 1, independent of $\boldsymbol{V}_1, \boldsymbol{V}_2, \cdots$, and define

$$\Gamma_k = \sum_{i=1}^{k} E_i \qquad\qquad \boldsymbol{X}_n = c(\alpha) \sum_{k=1}^{n} \Gamma_k^{-1/\alpha}\, \boldsymbol{V}_k \tag{22}$$

where

$$c(\alpha) = \left(\frac{\kappa}{\Lambda(\mathbb{S}^{d-1})}\right)^{-\frac{1}{\alpha}} \& \kappa = \begin{cases} \frac{\Gamma(2-\alpha)\cos(\frac{\pi\alpha}{2})}{1-\alpha} & \alpha \neq 1 \\ \frac{\pi}{2} & \alpha = 1. \end{cases} \tag{23}$$

- For $0 < \alpha < 1$, $\boldsymbol{X}_n$ converges in total variation to a stable vector $\boldsymbol{X}$ with spectral measure $\Lambda$ and stability index $\alpha$.
- For $1 \leq \alpha < 2$, whenever

$$\int_{\mathbb{S}^{d-1}} s\Lambda(ds) = 0, \tag{24}$$

the random vector $\boldsymbol{X}_n$ converges in total variation to a stable vector $\boldsymbol{X}$ with spectral measure $\Lambda$ and stability index $\alpha$.

Whenever there is convergence to $\boldsymbol{X}$, we write

$$\boldsymbol{X} = c(\alpha) \sum_{k=1}^{\infty} \Gamma_k^{-1/\alpha}\, \boldsymbol{V}_k. \tag{25}$$

Theorem 1 indicates that sampling a multivariate stable vector can be done by computing the truncated series as in equation (20). Since the vector $\boldsymbol{X}$ resulting from the series has a zero shift vector $\boldsymbol{\mu}^0 = \mathbf{0}$ V. Bentkus, A. Juozulynas, and V. Paulauskas [17, Remark 1], we simply add $\boldsymbol{\mu}^0$ after sampling $\boldsymbol{X}_n$ using the LePage series, when needed. The analysis of the truncation error is deferred to Appendix in section ??.

In the following section, we present algorithms to sample according to well-known spectral measures, namely: Isotropic, Sub-Gaussian, Discrete, and Mixed spectral measures.

## *4.2. Common spectral measures*

### 4.2.1. *Discrete spectral measures*

In the case of a discrete spectral measure with $M$ mass points, we store the point masses in an array $[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_M]$ and draw randomly with the probability of each vector $\boldsymbol{v}_i$ being

$$\Pr(\boldsymbol{v}_i) = \frac{\Lambda(\boldsymbol{v}_i)}{\sum_{\ell=1}^{M} \Lambda(\boldsymbol{v}_\ell)}. \tag{26}$$

### 4.2.2. *Isotropic spectral measures*

For an isotropic spectral measure, we sample the vectors $\{\boldsymbol{V}_k\}_{1 \leq k \leq n}$ uniformly on $\mathbb{S}^{d-1}$. To this end, several algorithms are available and we refer the reader to S. Schnabel and W. Janke [18] where various such algorithms are discussed. In our computations, we adopt the method presented in M. E. Muller [19] due to its simplicity and computational efficiency. Our

package sets the spectral measure of the sphere to 1, and then scales the truncated LePage series by an appropriate factor $q$ to adjust for the desired scale $\gamma$ given by the user: When the spectral measure of the sphere is 1, it can be verified that the characteristic function is given by

$$\phi_{\boldsymbol{X}}(\boldsymbol{t}) = \exp\left\{-\frac{\Gamma\left(\frac{d}{2}\right)}{\sqrt{\pi}}\frac{\Gamma\left(\frac{\alpha+1}{2}\right)}{\Gamma\left(\frac{\alpha+d}{2}\right)} \mid \boldsymbol{t} \mid^{\alpha}\right\}. \tag{27}$$

To achieve a scale $\gamma$, we therefore multiply the truncated series by

$$q = \gamma\left[\frac{\Gamma\left(\frac{d}{2}\right)}{\sqrt{\pi}}\frac{\Gamma\left(\frac{\alpha+1}{2}\right)}{\Gamma\left(\frac{\alpha+d}{2}\right)}\right]^{-\frac{1}{\alpha}}. \tag{28}$$

#### 4.2.3. *Sub-Gaussian (elliptical) spectral measures*

A sub-Gaussian random vector is characterized by a positive-definite $d \times d$ shape matrix $\Sigma$. If $\boldsymbol{X}$ is a unit-scale isotropic multivariate stable random vector with stability index $\alpha$ then $\boldsymbol{Y} = \Sigma^{\frac{1}{2}}\boldsymbol{X}$ is a sub-Gaussian random vector with shape matrix $\Sigma$ where $\Sigma^{\frac{1}{2}}$ is the Cholesky decomposition of $\Sigma$ J. P. Nolan [4]. As such, to sample from $\boldsymbol{Y}_n$, one can first sample a unit-scale isotropic random vector with the same stability index $\alpha$ using the LePage series as in Section 7.2 and multiply it by the Cholesky decomposition $\Sigma^{\frac{1}{2}}$ of $\Sigma$. In section 11 in the appendix, we analyze the resulting Mean Square Error (MSE) of $\boldsymbol{Y}_n$.

A useful alternative would be to sample the vectors $\boldsymbol{V}_k$ for the sub-Gaussian case by sampling vectors $\boldsymbol{U}_k$ uniformly distributed on the unit sphere and multiplying them by $\Sigma^{\frac{1}{2}}$. Indeed, if $\boldsymbol{X}_n$ is the $n$-terms LePage series, by Theorem 1, $\boldsymbol{X}_n$ converges in total variation to $\boldsymbol{X}$, and

$$\boldsymbol{Y}_n = \Sigma^{\frac{1}{2}}\boldsymbol{X}_n = c(\alpha)\Sigma^{\frac{1}{2}}\sum_{k=1}^{n}\Gamma_k^{-\frac{1}{\alpha}}\boldsymbol{U}_k = c(\alpha)\sum_{k=1}^{n}\Gamma_k^{-\frac{1}{\alpha}}\boldsymbol{V}_k \qquad \text{where } \boldsymbol{V}_k = \Sigma^{\frac{1}{2}}\boldsymbol{U}_k, \tag{29}$$

converges in total variation to $\boldsymbol{Y} = \Sigma^{\frac{1}{2}}\boldsymbol{X}$, which is the desired sub-Gaussian random vector.

The package first samples $\boldsymbol{V}_k$ and then computes the sum presented in (25) to sample a sub-Gaussian random vector $\boldsymbol{Y}_n$.

#### 4.2.4. *Mixed spectral measures*

Now consider the scenario where the spectral measure at hand is a mixture of $L$ different spectral measures: $\Lambda_1, \Lambda_2, \cdots, \Lambda_L$. If $w_1, w_2, \cdots, w_L$ are their respective associated weights, the mixed spectral measure $\Lambda_{\text{mix}}$ is given by

$$\Lambda_{\text{mix}} = \sum_{i=1}^{L} w_i\Lambda_i. \tag{30}$$

To sample from this spectral measure, we proceed as follows.

- For each weight $w_i$ we associate its corresponding probability as

$$p_i = \frac{w_i}{\sum_{\ell=1}^{L} w_\ell} \qquad i = 1, \cdots, L. \tag{31}$$

Then, we sort the probabilities in non-decreasing order that we assume –without loss of generality– to be $p_1, \cdots, p_L$ hereafter.

- We draw $U \sim \mathcal{U}[0,1]$,
  - If $U \leq p_1$ sample from $\Lambda_1$,
  - else if $U \leq p_1 + p_2$, sample from $\Lambda_2$,
  - else iterate on the remaining spectral measures.

This algorithm achieves a sampling from the mixture of $\Lambda_1, \cdots, \Lambda_L$ with probabilities $p_1, \cdots, p_L$ respectively.

## 5. Implementation

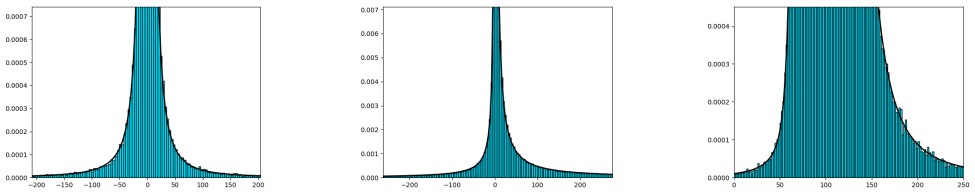
Whenever the LePage series is used, the package chooses the number of terms to achieve an MSE less than 0.01 using Theorem 2 in the Appendix in section ??. This number is capped at 50,000 terms and a warning is issued whenever this cap is reached.

### *5.1. Univariate random number generator*

In the univariate case, the software samples $X \sim \mathcal{S}(\alpha, \beta, \gamma, \delta; k)$, and the type 1 parameterization is used by default.

```
import aub_htp as ht
alpha=1
beta=0
loc=0
scale=1
n=500000
samples=ht.alpha_stable.rvs(alpha=alpha, beta=beta, loc=loc, scale=scale, size=n)
```

In Figure 4, we present some examples where univariate $\alpha$-stable variables were sampled using our implementation, and plot the truncated histogram of values. Each histogram is overlayed by the pdf values which are computed using the SciPy `levy_stable` numerical package.

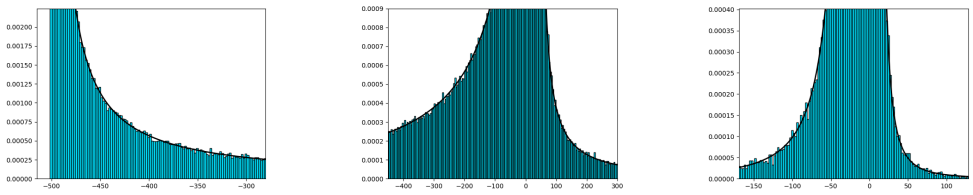


(a) (b) (c)

**Figure 4**. *Truncated histogram of $\alpha$-stable distributions, each containing $5 \times 10^5$ points.*

An example using the type 0 parameterization is presented in Figure 5.

```python
import aub_htp as ht

alpha = 0.1
beta = 1
loc = -500
scale = 10
n = 500000
ht.alpha_stable.with_parameterization("S0")
samples = ht.alpha_stable.rvs(alpha=alpha, beta=beta, loc=loc, scale=scale, size=n)
```



(a) (b) (c)

**Figure 5**. *Truncated histogram of $\alpha$-stable distributions, each containing $5 \times 10^5$ points.*

## *5.2. Multivariate random number generator*

### *Isotropic $\alpha$-stable random vector:*

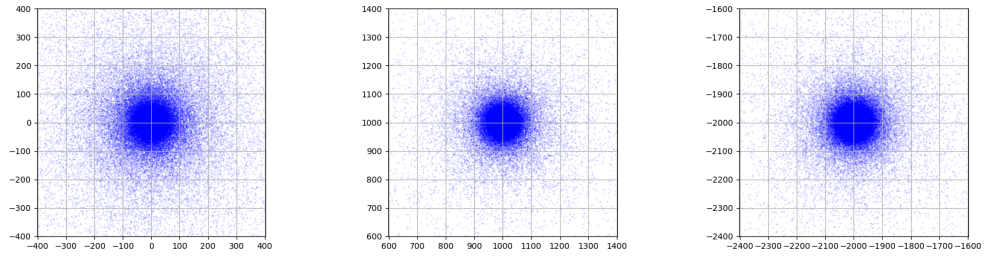
An isotropic $\alpha$-stable random vector with characteristic function

$$\phi_{\boldsymbol{X}}(\boldsymbol{t}) = e^{-\gamma^{\alpha} \mid \boldsymbol{t} \mid^{\alpha} + \mathrm{i}\boldsymbol{t}^{T}\boldsymbol{\mu}^{0}}, \tag{32}$$

can be generated using the AUB-HTP package as follows:

```python
import aub_htp as ht
from aub_htp.random import IsotropicSampler

alpha = 0.5
gamma = 1
shift = [0, 0]
d = 2
n = 500000
sampler = IsotropicSampler(number_of_dimensions=d, alpha=alpha, gamma=gamma)
samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=sampler, shift=shift, size=n
)
```

where $\alpha$ is the stability index, $\gamma$ is the scale parameter and $\boldsymbol{\mu}^0$ represents the shift vector. Figure 6 depicts scatter plots of the generated isotropic $\alpha$-stable random vectors.



(a) (b) (c)

**Figure 6**. *Scatter plots each containing $5 \times 10^5$ points of isotropic $\alpha$-stable distributions. The plots have been truncated to a range of 400 points within the shift vector.*

*Elliptic $\alpha$-stable random vector:*

An elliptic $\alpha$-stable random vector with shape matrix $\Sigma$ with the following characteristic function

$$\phi_{\boldsymbol{X}}(\boldsymbol{t}) = e^{-(\boldsymbol{t}^T\Sigma\boldsymbol{t})^{\frac{\alpha}{2}} + i\boldsymbol{t}^T\boldsymbol{\mu}^0}, \tag{33}$$

can also be generated using AUB-HTP. We note that the elliptic sampler takes the total mass of the unit sphere as input. If the total mass of the sphere is not specified, the software estimates it using Monte Carlo simulations (with $10^6$ samples) as follows:

$$\Lambda\left(\mathbb{S}^{d-1}\right) = \mathbb{E}\left[\mid \Sigma^{\frac{1}{2}}\boldsymbol{U} \mid^{\alpha}\right] \approx \frac{1}{n}\sum_{i=1}^{n} \mid \Sigma^{\frac{1}{2}}\boldsymbol{U}_i \mid^{\alpha}, \tag{34}$$

where the equality is due to G. Samorodnitsky and M. S. Taqqu [13, proposition 2.5.8], and where $\boldsymbol{U}_1, \boldsymbol{U}_2, ...\boldsymbol{U}_n$ are i.i.d. samples according to $\boldsymbol{U} \sim \mathcal{U}\left(\mathbb{S}^{d-1}\right)$. Equation (30) is justified by the application of the Weak Law of Large Numbers (WLLN) because the random variable $\mid \Sigma^{\frac{1}{2}}\boldsymbol{U} \mid^{\alpha}$ has a finite second moment. Indeed,

$$\begin{aligned}
\mathbb{E}\left[\mid \Sigma^{\frac{1}{2}}\boldsymbol{U} \mid^{2\alpha}\right] &= \mathbb{E}\left[\left(\boldsymbol{U}^T\Sigma^{\frac{1}{2}T}\Sigma^{\frac{1}{2}}\boldsymbol{U}\right)^{\alpha}\right] \\
&= \mathbb{E}\left[\left(\boldsymbol{U}^T\Sigma^T\boldsymbol{U}\right)^{\alpha}\right] \\
&= \mathbb{E}[\mid \boldsymbol{U} \mid_{\Sigma}^{2\alpha}] \\
&\leq \lambda_{\max}^{\alpha}(\Sigma)\mathbb{E}[\mid \boldsymbol{U} \mid^{2\alpha}] = \lambda_{\max}^{\alpha}(\Sigma) < \infty,
\end{aligned} \tag{35}$$

where equality $(a)$ is due to the Cholesky decomposition definition and in $(b)$ we used the $\Sigma$-weighted norm $\mid \boldsymbol{x} \mid_{\Sigma}^2 \triangleq \boldsymbol{x}^T\Sigma\boldsymbol{x}$. Finally, the inequality in $(c)$ is due to the fact that $\mid \boldsymbol{x} \mid_{\Sigma}^2 \leq \lambda_{\max}(\Sigma) \mid \boldsymbol{x} \mid^2$ where $\lambda_{\max}(\Sigma)$ is the largest eigenvalue of $\Sigma$, equal to the norm of matrix $\Sigma$ R. A. Horn and C. R. Johnson [20].

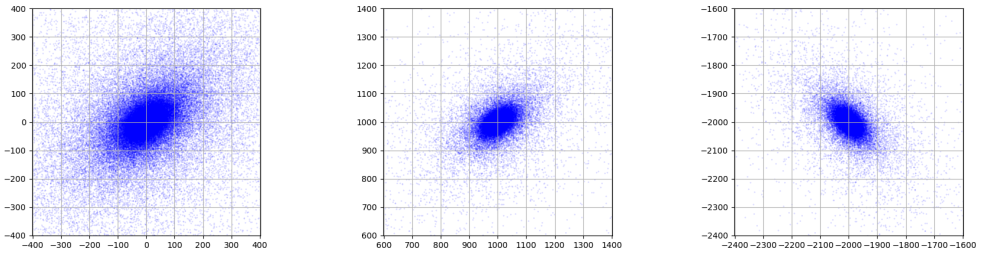
Finally, we note that the MSE resulting from estimating the total mass of the sphere using equation (30) is given by H. Niederreiter [21, Theorem 1.1].

An example code is as follows.

```
import aub_htp as ht
from aub_htp.random import EllipticSampler
alpha = 0.5
shift= [0,0]
d = 2
n=500000
sigma1=[[2,0.8],[0.8,1.5]]
sampler =EllipticSampler(number_of_dimensions = 2, alpha = alpha, sigma = sigma1)
samples = ht.multivariate_alpha_stable.rvs(alpha = alpha, spectral_measure_sampler =
sampler,shift=shift, size = n)
```

Figure 7 shows the scatter plots of the elliptical $\alpha$-stable random vector, where we use the following shape matrices

$$\Sigma_1 = \begin{pmatrix} 2 & 0.8 \\ 0.8 & 1.5 \end{pmatrix} \qquad \Sigma_2 = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}. \tag{36}$$



(a) (b) (c)

**Figure 7**. *Scatter plots each containing $5 \times 10^5$ points of elliptic contoured $\alpha$-stable distributions; the plots have been truncated to a range of 400 points within the shift vector.*

### *Discrete spectral measures:*

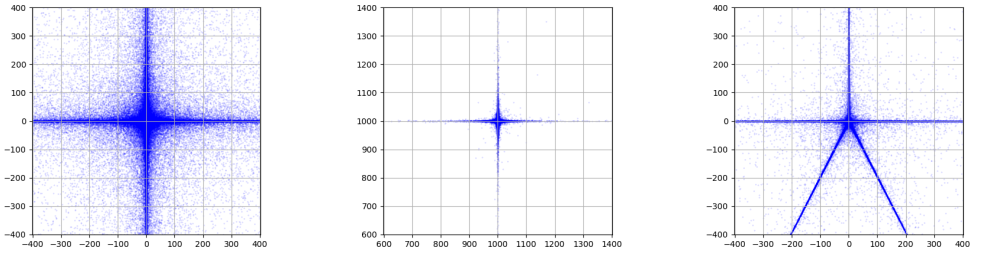
For an $\alpha$-stable random vector that has a discrete spectral measure with point masses at positions $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_M$ and corresponding weights $w_1, \cdots, w_M$, the corresponding characteristic function is given by

$$\phi_{\boldsymbol{X}}(\boldsymbol{t}) = \exp\left\{ -\sum_{\ell=1}^{M} w_\ell \psi_\alpha(\boldsymbol{t}^T \boldsymbol{v}_\ell) + \mathrm{i}\boldsymbol{t}^T \boldsymbol{\mu}^0 \right\}. \tag{37}$$

Such a vector can be generated as follows:

```
import aub_htp as ht
from aub_htp.random import DiscreteSampler

alpha = 0.2
shift = [0, 0]
d = 2
n = 500000
positions = [[1, 0], [0, 1], [-1, 0], [0, -1]]
weights = [1, 1, 1, 1]
sampler = DiscreteSampler(alpha=alpha, positions=positions, weights=weights)
samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=sampler, shift=shift, size=n
)
```

with corresponding scatter plots shown in Figure 8.



(a) (b) (c)

**Figure 8**. *Scatter plots each containing $5 \times 10^5$ points of $\alpha$-stable distributions with a discrete spectral measure. The plots have been truncated to a range of 400 points within the shift vector.*

### *Mixed spectral measure:*

For a mixed spectral measure, let $\Lambda_1, \cdots, \Lambda_L$ be $L$ spectral measures with associated weights $w_1, \cdots, w_L$, and consider their mixture $\Lambda_{\text{mix}} = \sum_{i=1}^{L} w_i \Lambda_i$. To sample according to this spectral measure, we proceed as follows:

```python
import aub_htp as ht
from aub_htp.random import DiscreteSampler, IsotropicSampler, MixedSampler

alpha = 0.3
shift = [1000, 1000]
d = 2
n = 500000
gamma = 1

# Discrete Spectral Measure
positions = [[1, 0], [0, 1], [-1, 0], [0, -1]]
weights = [1, 1, 1, 1]
sampler1 = DiscreteSampler(alpha=alpha, positions=positions, weights=weights)

# Isotropic Spectral Measure
sampler2 = IsotropicSampler(number_of_dimensions=d, alpha=alpha, gamma=gamma)

# Spectral Measures and Their Associated Weights
spectral_measures = [sampler1, sampler2]
measure_weights = [0.5, 0.5]

mixed_sampler = MixedSampler(
    spectral_measures=spectral_measures, weights=measure_weights
)
samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=mixed_sampler, shift=shift, size=n
)
```

where `spectral_measures` is a list containing the spectral measures that we are mixing, and `measure_weights` contains the associated weight of each measure. The resulting scatter plot is shown in Figure 9a.

*Custom spectral measures:*

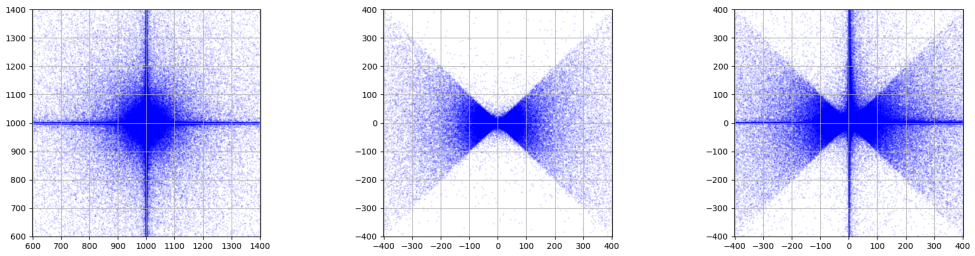
The common spectral measures presented above are built-in into AUB-HTP. If the user wants to sample from an $\alpha$-stable random vector whose spectral measure is not built-in, this is made possible using a systematic procedure outlined in the following example:

- Say the user is interested in sampling a multivariate $\alpha$-stable random vector in dimension $d = 2$ with $\alpha = 0.3, \boldsymbol{\mu}^0 = \mathbf{0}$, and whose spectral measure is uniform over the arcs $[-\frac{\pi}{4}, \frac{\pi}{4}]$ and $[\frac{3\pi}{4}, \frac{5\pi}{4}]$, that is, with probability 0.5, we draw a point uniformly on the arc $[\frac{3\pi}{4}, \frac{5\pi}{4}]$, and with probability 0.5, we draw a point uniformly on the arc $[-\frac{\pi}{4}, \frac{\pi}{4}]$.
- We assign equal probabilities to both arcs and the spectral measure of each arc is half the total measure of the sphere. Generally, the probability of each region is the ratio of the spectral measure of the region to the total mass of the sphere.
- We create a class of the spectral measure sampler that we want by inheriting from the base class `BaseSpectralMeasureSampler`. In the class that the user is creating, the function which samples the vectors $\boldsymbol{V}$ in Theorem 1 should be provided. Moreover, the dimension of the desired alpha-stable vector and the total mass of the unit sphere should be specified. The created spectral measure sampler is then passed as an argument to the `rvs()` method as shown in the following code.

The steps are presented below.

```python
import aub_htp as ht
from aub_htp.random import BaseSpectralMeasureSampler

alpha = 0.3
shift = [0, 0]
d = 2
n = 500000


class ButterflySampler(BaseSpectralMeasureSampler):
    def sample(self, number_of_samples: int, random_state=None):
        p = np.random.rand(number_of_samples)
        theta = np.empty(number_of_samples)

        mask = p <= 0.5
        theta[mask] = np.random.uniform(-np.pi / 4, np.pi / 4, size=mask.sum())
        theta[~mask] = np.random.uniform(
            3 * np.pi / 4, 5 * np.pi / 4, size=(~mask).sum()
        )

        x = np.cos(theta)
        y = np.sin(theta)

        return np.column_stack((x, y))

    def dimensions(self) -> int:
        return 2

    def mass(self) -> float:
        return 1.0


samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=ButterflySampler(), size=n, shift=shift
)
```

We highlight that the mass of the sphere and the dimension should be specified when implementing the sub-class. The resulting scatter plot is shown in Figure 9b. It is important to emphasize that when $\alpha \geq 1$, the custom spectral measure should satisfy the condition in equation (22).



(a) (b) (c)

**Figure 9**. *Scatter plots showing a custom spectral measure and different mixtures of spectral measures. The plots have been truncated to a range of 400 points within the shift vector.*

One could mix a custom spectral measure, with another spectral measure. For example, we mix below equally the spectral measure defined above with the discrete spectral measure with $\text{masses} = [(1,0),(0,1),(-1,0),(0,-1)]$ and $\text{weights} = [1,1,0.25,0.25]$, and the random vector has $\alpha = 0.25$, $\boldsymbol{\mu}^{0T} = (0,0)$.

```python
import aub_htp as ht
from aub_htp.random import BaseSpectralMeasureSampler, DiscreteSampler, MixedSampler

alpha = 0.25
shift = [0, 0]
d = 2
n = 1000000


class ButterflySampler(BaseSpectralMeasureSampler):
    def sample(self, number_of_samples: int, random_state=None):
        p = np.random.rand(number_of_samples)
        theta = np.empty(number_of_samples)

        mask = p <= 0.5
        theta[mask] = np.random.uniform(-np.pi / 4, np.pi / 4, size=mask.sum())
        theta[~mask] = np.random.uniform(
            3 * np.pi / 4, 5 * np.pi / 4, size=(~mask).sum()
        )

        x = np.cos(theta)
        y = np.sin(theta)

        return np.column_stack((x, y))

    def dimensions(self) -> int:
        return 2

    def mass(self) -> float:
        return 1.0


samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=ButterflySampler(), size=n, shift=shift
)

# Discrete Spectral Measure
positions = [[1, 0], [0, 1], [-1, 0], [0, -1]]
weights = [1, 1, 0.25, 0.25]
sampler1 = DiscreteSampler(alpha=alpha, positions=positions, weights=weights)

# Spectral Measures and Their Associated Weights
spectral_measures = [sampler1, ButterflySampler()]
measure_weights = [0.5, 0.5]

mixed_sampler = MixedSampler(
    spectral_measures=spectral_measures, weights=measure_weights
)
samples = ht.multivariate_alpha_stable.rvs(
    alpha=alpha, spectral_measure_sampler=mixed_sampler, shift=shift, size=n
)
```

The resulting scatterplot is shown in Figure 9c.

## 6. Conclusion

This paper presented AUB-HTP, a Python package for density computation and random variate generation of $\alpha$-stable distributions. The package is motivated by the practical difficulty of working with $\alpha$-stable laws, whose densities generally lack closed-form expressions and whose multivariate simulation requires careful handling of the underlying spectral measure.

AUB-HTP provides scalar density evaluation through a hybrid strategy that combines input standardization, Zolotarev-type integral representations, series formulas, and numerical inversion of characteristic functions. This design improves robustness across parameter regimes and enables systematic comparisons in terms of accuracy and runtime. The validation results show that the package provides reliable scalar density values and identify

parameter regimes in which SciPy's reference implementation appears to suffer from numerical inaccuracies.

The package also implements scalar and multivariate random variate generation. For multivariate $\alpha$-stable vectors, AUB-HTP uses LePage series representations and supports several spectral-measure models, including discrete, isotropic, sub-Gaussian/elliptical, mixed, and custom spectral measures. These features extend the range of $\alpha$-stable models that can be simulated within the Python ecosystem. The truncation-error analysis further provides guidance on the approximation induced by finite LePage-series representations.

By bringing together density evaluation, flexible simulation, validation, runtime analysis, and extensible implementation tools, AUB-HTP aims to make $\alpha$-stable modeling more accessible for computational work involving heavy-tailed data. Future developments may include multivariate PDF computation, CDF evaluation, parameter-estimation routines, further numerical acceleration, and application-oriented modules for communications, signal processing, and machine learning.

## 7. Appendix: Error analysis for the LePage series estimate

In this appendix, we derive an upper bound on the mean square error of the LePage series estimate; we frequently use the following asymptotics of the gamma function from [22]: Let $x, a, b \in \mathbb{R}$, then we have

$$\lim_{x\to\infty} \frac{\Gamma(x+a)}{\Gamma(x+b)\, x^{a-b}} = 1, \tag{38}$$

then $\forall a, b \in \mathbb{R}$, there exists an $n_0$ such that

$$\frac{\Gamma(k+a)}{\Gamma(k+b)} < 2k^{a-b} \qquad \forall k \geq n_0. \tag{39}$$

### *7.1. The LePage series truncation error*

The proof is as follows.

We start with the case $\alpha < 1$. We have

$$\mid \boldsymbol{X}_n - \boldsymbol{X} \mid = \mid c(\alpha) \sum_{k=n+1}^{\infty} \Gamma_k^{-\frac{1}{\alpha}} \boldsymbol{V}_k \mid \leq c(\alpha) \sum_{k=n+1}^{\infty} \mid \Gamma_k^{-\frac{1}{\alpha}} \boldsymbol{V}_k \mid = c(\alpha) \sum_{k=n+1}^{\infty} \Gamma_k^{-\frac{1}{\alpha}}, \tag{41}$$

which implies

**Theorem 2.**

Let $\boldsymbol{X}$ and $\boldsymbol{X}_n$ be as in Theorem 1, and assume in addition that $\mid \boldsymbol{V}_k \mid = 1$ for all $k = 1, \cdots, n$. Let $n_0$ be the same as in (35) then $\forall n \geq \max\{n_0, \frac{2}{\alpha}\}$, $\mathbb{E} \mid \boldsymbol{X}_n - \boldsymbol{X} \mid^2 \leq \delta_n$ where

$$\delta_n = \begin{cases} \frac{2\alpha^2}{(1-\alpha)^2} c^2(\alpha) n^{2-\frac{2}{\alpha}} \alpha < 1 \\ \frac{2\alpha}{2-\alpha} c^2(\alpha) n^{1-\frac{2}{\alpha}} \quad \alpha \geq 1. \end{cases} \tag{40}$$

$$
\begin{aligned}
\mathbb{E}[|\ \boldsymbol{X}_n - \boldsymbol{X}\ |^2] &\leq \mathbb{E}\left[\left(c(\alpha)\sum_{k=n+1}^{\infty}\Gamma_k^{-\frac{1}{\alpha}}\right)^2\right] = c^2(\alpha)\left(\sum_{k=n+1}^{\infty}\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right] + 2\sum_{n<i<j}\mathbb{E}\left[\Gamma_i^{-\frac{1}{\alpha}}\Gamma_j^{-\frac{1}{\alpha}}\right]\right) \\
&\overset{(a)}{\leq} c^2(\alpha)\left(\sum_{k=n+1}^{\infty}\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right] + 2\sum_{n<i<j}\sqrt{\mathbb{E}\left[\Gamma_i^{-\frac{2}{\alpha}}\right]\mathbb{E}\left[\Gamma_j^{-\frac{2}{\alpha}}\right]}\right) \\
&= c^2(\alpha)\left(\sum_{k=n+1}^{\infty}\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right] + \left(\sum_{k=n+1}^{\infty}\sqrt{\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right]}\right)^2 - \sum_{k=n+1}^{\infty}\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right]\right) \\
&\overset{(b)}{=} c^2(\alpha)\left(\sum_{k=n+1}^{\infty}\sqrt{\frac{\Gamma(k-\frac{2}{\alpha})}{\Gamma(k)}}\right)^2 \\
&\overset{(c)}{<} c^2(\alpha)\left(\sum_{k=n+1}^{\infty}\sqrt{2k^{-\frac{2}{\alpha}}}\right)^2 \\
&= 2c^2(\alpha)\left(\sum_{k=n+1}^{\infty}k^{-\frac{1}{\alpha}}\right)^2 \\
&\overset{(d)}{\leq} \frac{2\alpha^2}{(1-\alpha)^2}c^2(\alpha)n^{2-\frac{2}{\alpha}}
\end{aligned}
\tag{42}
$$

where inequality $(a)$ is due to the Cauchy-Schwarz inequality. The equality in $(b)$ is due to the fact that $\Gamma_k \sim \text{Gamma}(k,1), n \geq \frac{2}{\alpha}$, and $\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right] = \frac{\Gamma(k-\frac{2}{\alpha})}{\Gamma(k)}$, and inequality $(c)$ is due to (35). Inequality $(d)$ is due to the fact that

$$
\sum_{k=n+1}^{\infty}k^{-\frac{1}{\alpha}} \leq \int_n^{\infty}x^{-\frac{1}{\alpha}}dx = \frac{\alpha}{1-\alpha}n^{1-\frac{1}{\alpha}}. \tag{43}
$$

For the case $\alpha \geq 1$, since $|\ \boldsymbol{V}_k\ | = 1\ \forall\ k = 1, \cdots, n$, equation (22) implies $\mathbb{E}[\boldsymbol{V}] = \boldsymbol{0}$. Let $\boldsymbol{V}_j = \left(v_j^1, v_j^2, \cdots, v_j^d\right)$ then we have

$$
\begin{aligned}
\mathbb{E}[|\ \boldsymbol{X}_n - \boldsymbol{X}\ |^2] &= c^2(\alpha)\,\mathbb{E}\left[\left\langle\sum_{\ell=n+1}^{\infty}\Gamma_k^{-\frac{1}{\alpha}}\boldsymbol{V}_\ell, \sum_{k=n+1}^{\infty}\Gamma_k^{-\frac{1}{\alpha}}\boldsymbol{V}_k\right\rangle\right] \\
&= c^2(\alpha)\mathbb{E}\left[\sum_{k=n+1}^{\infty}\sum_{\ell=n+1}^{\infty}\sum_{i=1}^{d}\Gamma_k^{-\frac{1}{\alpha}}\Gamma_\ell^{-\frac{1}{\alpha}}v_k^i v_\ell^i\right] \\
&\overset{(a)}{=} c^2(\alpha)\left[\sum_{k=n+1}^{\infty}\sum_{\ell=n+1}^{\infty}\sum_{i=1}^{d}\mathbb{E}\left[\Gamma_k^{-\frac{1}{\alpha}}\Gamma_\ell^{-\frac{1}{\alpha}}\right]\mathbb{E}\left[v_k^i v_\ell^i\right]\right]] \\
&\overset{(b)}{=} c^2(\alpha)\sum_{k=n+1}^{\infty}\mathbb{E}\left[\Gamma_k^{-\frac{2}{\alpha}}\right]\mathbb{E}[|\ \boldsymbol{V}_k\ |^2] \\
&= c^2(\alpha)\sum_{k=n+1}^{\infty}\frac{\Gamma(k-\frac{2}{\alpha})}{\Gamma(k)} < 2c^2(\alpha)\sum_{k=n+1}^{\infty}k^{-\frac{2}{\alpha}} \\
&\leq \frac{2\alpha}{2-\alpha}c^2(\alpha)n^{1-\frac{2}{\alpha}},
\end{aligned}
\tag{44}
$$

where equality $(a)$ is due to the fact that $\boldsymbol{V}_k \perp\!\!\!\perp \Gamma_k$ and $(b)$ is justified since $v_k^i \perp\!\!\!\perp v_\ell^i, \forall\ k \neq \ell$ and $\mathbb{E}[v_k^i] = 0, \forall\ k$. The rest of the equalities and inequalities follow by similar steps as in the case $\alpha < 1$.

One can clearly see that the bound on the error decays faster for smaller values of $\alpha$. This indicates that this sampling method can appropriately handle the difficult region of small values of $\alpha$.

### *7.2. Sub-Gaussian error*

Sampling a multivariate sub-Gaussian $\alpha$-stable vector can be done by first computing the truncated series $\boldsymbol{X}_n$ for the isotropic spectral measure with unit-scale, and then applying $\boldsymbol{Y}_n = \Sigma^{\frac{1}{2}}\boldsymbol{X}_n$ where $\Sigma^{\frac{1}{2}}$ is the Cholesky decomposition of $\Sigma$– the desired shape matrix as described in Section 7.3. The resulting MSE in that case satisfies

$$\begin{aligned}\mathbb{E}[\mid \boldsymbol{Y}_n - \boldsymbol{Y} \mid^2] &= \mathbb{E}\Big[\mid \Sigma^{\frac{1}{2}}(\boldsymbol{X}_n - \boldsymbol{X}) \mid^2\Big] \\ &= \mathbb{E}\Big[(\boldsymbol{X}_n - \boldsymbol{X})^T \Sigma^{\frac{1}{2}T}\Sigma^{\frac{1}{2}}(\boldsymbol{X}_n - \boldsymbol{X})\Big] = \mathbb{E}\Big[(\boldsymbol{X}_n - \boldsymbol{X})^T \Sigma(\boldsymbol{X}_n - \boldsymbol{X})\Big] \\ &\leq \lambda_{\max}(\Sigma)\,\mathbb{E}[\mid \boldsymbol{X}_n - \boldsymbol{X} \mid^2],\end{aligned} \tag{45}$$

where we used the fact that $\mid \boldsymbol{x} \mid^2_{\Sigma} \leq \lambda_{\max}(\Sigma) \mid \boldsymbol{x} \mid^2$, with $\lambda_{\max}(\Sigma)$ –the largest eigenvalue of $\Sigma$ – being the norm of matrix $\Sigma$ R. A. Horn and C. R. Johnson [20]. Finally, one can upperbound $\mathbb{E}[\mid \boldsymbol{X}_n - \boldsymbol{X} \mid^2]$ using Theorem 2.

## REFERENCES


[1] M. Nassar, K. Gulati, A. Sujeeth, N. Aghasadeghi, B. Evans, and K. Tinsley, "Mitigating near-field interference in laptop embedded wireless transceivers," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, Mar. 2008, pp. 1405–1408. doi: 10.1109/ICASSP.2008.4517882.

[2] G. Tsihrintzis, P. Tsakalides, and C. Nikias, "Signal detection in severely heavy-tailed radar clutter," in *Conference Record of The Twenty-Ninth Asilomar Conference on Signals, Systems and Computers*, 1995, p. 865–869 vol.2. doi: 10.1109/ACSSC.1995.540823.

[3] M. E. Crovella, M. S. Taqqu, and A. Bestavros, "Heavy-tailed probability distributions in the World Wide Web," *A practical guide to heavy tails*, vol. 1, pp. 3–26, 1998.

[4] J. P. Nolan, "Multivariate elliptically contoured stable distributions: theory and estimation," *Computational Statistics*, vol. 28, no. 5, pp. 2067–2089, 2013, doi: 10.1007/s00180-013-0396-7.

[5] N. N. Taleb, "Statistical Consequences of Fat Tails: Real World Preasymptotics, Epistemology, and Applications." [Online]. Available: https://arxiv.org/abs/2001.10488

[6] B. V. Gnedenko and A. N. Kolmogorov, *Limit Distributions for Sums of Independent Random Variables*. Reading Massachusetts: Addison-Wesley Publishing Company, 1968.

[7] J. M. Chambers, C. L. Mallows, and B. W. Stuck, "A Method for Simulating Stable Random Variables," *Journal of the American Statistical Association*, vol. 71, no. 354, pp. 340–344, 1976.

[8] V. M. Zolotarev, *One-Dimensional Stable Distributions*, vol. 65. in Translations of Mathematical Monographs, vol. 65. Providence, RI: American Mathematical Society, 1986.

[9] A. V. Skorohod, "Asymptotic formulas for stable distribution laws," *Selected translations in mathematical statistics and probability*, vol. 1, pp. 157–161, 1961.

[10] H. Pollard, "The Representation of \(e^-x^\lambda\) as a Laplace Integral", *Bulletin of the American Mathematical Society*, vol. 52, no. 10, pp. 908–910, 1946, doi: 10.1090/S0002-9904-1946-08672-3.

[11] H. Bergström, "On Some Expansions of Stable Distribution Functions," *Arkiv för Matematik*, vol. 2, no. 4, pp. 375–378, 1952, doi: 10.1007/BF02591503.

[12] R. LePage, M. Woodroofe, and J. Zinn, "Convergence to a Stable Distribution Via Order Statistics," *The Annals of probability*, vol. 9, no. 4, pp. 624–632, 1981.

[13] G. Samorodnitsky and M. S. Taqqu, *Stable non-Gaussian random processes : stochastic models with infinite variance*, 1st ed. Boca Raton: Chapman & Hall/CRC, 2000.

[14] J. P. Nolan, *Univariate Stable Distributions: Models for Heavy Tailed Data*, 1st Edition 2020. in Springer Series in Operations Research and Financial Engineering. Cham: Springer Nature, 2020.

[15] A. Wintner, "The Singularities of Cauchy's Distributions," *Duke Mathematical Journal*, vol. 8, pp. 678–681, 1941.

[16] The SciPy Community, "scipy.stats.levy_stable," technical report, 2026. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.levy_stable.html

[17] V. Bentkus, A. Juozulynas, and V. Paulauskas, "Lévy-LePage Series Representation of Stable Vectors: Convergence in Variation," *Journal of theoretical probability*, vol. 14, no. 4, pp. 949–978, 2001.

[18] S. Schnabel and W. Janke, "A simple algorithm for uniform sampling on the surface of a hypersphere," *arXiv.org*, 2022.

[19] M. E. Muller, "A note on a method for generating points uniformly on n -dimensional spheres," vol. 2. pp. 19–20, 1959.

[20] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge: Cambridge University Press, 1985.

[21] H. Niederreiter, *Random number generation and quasi-Monte Carlo methods*. in CBMS-NSF Regional Conference Series in Applied Mathematics. Philadelphia, Pa: Society for Industrial, 1992.

[22] "NIST Digital Library of Mathematical Functions." [Online]. Available: https://dlmf.nist.gov/