MATHÉMATIQUES
VISION
APPRENTISSAGE

A VAE Approach to Sample Multivariate Extremes

Extreme Values Theory

Khalil Braham, khalil.braham@telecom-paris.fr
Gabrielle Le Bellier, lebellier.gabrielle@gmail.com

Table des matières

1 Introduction

Extreme events, such as natural disasters and financial crashes, have a profound impact on society, causing significant damage to property and infrastructure, as well as loss of life. Therefore, it is essential to develop accurate methods for estimating the probability of these events. This information can be used to make informed decisions about risk mitigation and disaster preparedness.

Traditional methods for extreme value analysis are often limited to univariate data, while real-world data is often multivariate, meaning that it consists of multiple variables that are interrelated. For example, the occurrence of a flood may depend on factors such as rainfall, temperature, and snowpack. Therefore, it is important to develop methods for analyzing multivariate extreme events.

Extreme value theory (EVT) is a branch of statistics that deals with the modeling of extreme events. EVT provides a theoretical framework for understanding the behavior of extreme values, which can be used to estimate the probability of extreme events.

Variational autoencoders (VAEs) are a powerful machine learning technique for learning latent representations of data. VAEs can be used to learn a low-dimensional representation of high-dimensional data, which can be useful for tasks such as data compression and anomaly detection.

As extreme values theory tackles the question of rare events and machine learning principally focuses on global tendencies among data, it seemed like the two fields were difficult to combine. In this article, the authors bridged the gap between Extreme Values Theory and Machine Learning by adapting a VAE architecture to apply it to Extreme Values Theory.

Previous articles in the literature already focused on the application of some machine learning models, such as Generative Adversarial Networks (GANs) or Normalizing Flows (NFs), in order to sample extreme values distributions, especially heavy-tailed ones.

However, a recent idea emerged in the recent years : state-of-the-art models based on likelihood, such as VAEs seem better than GANs to sample heavy-tailed distributions.

In this article (1), the authors then wanted to use a VAE to sample such a distribution. After showing that a standard VAE was only able to sample light-tailed distribution, they proposed an adapted framework to sample a heavy-tailed distribution with VAEs.

2 Standard VAE

We recall in this section the principal of a VAE and present the standard VAE architecture.

2.1 Variational Auto Encoder

The goal of using a VAE is to generate samples from a random variable X, using a latent variable Z. It is using the following scheme :

1. We draw z from a prior random variable Z with a known probability density function $p_\alpha(z)$.
2. We sample x from the probability density function $p(x|z)$. As we do not know $p(x|z)$, we introduce the likelihood $p_\theta(x|z)$ where θ is a parameter to calibrate. To do so, we introduce a target distribution $q_\phi(z|x)$ parameterized by ϕ.

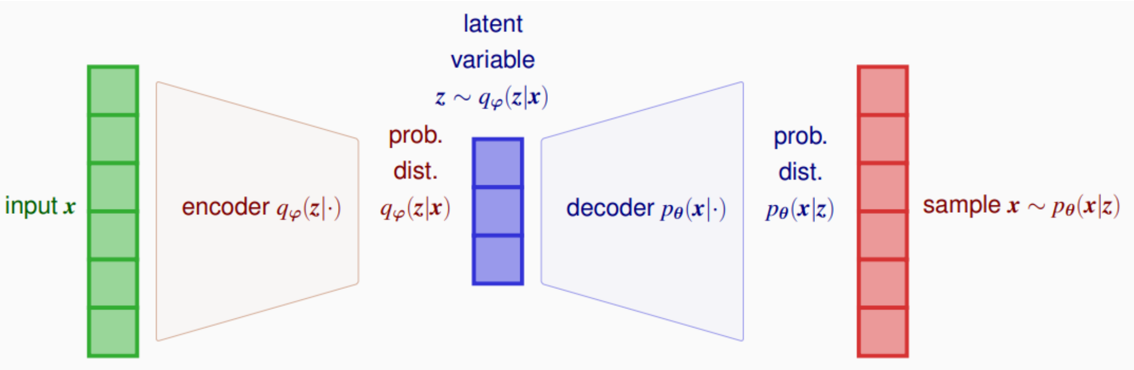


FIGURE 1 – Variational Autoencoder architecture

During the training of the model, we want to maximize the ELBO loss $L(x^{(i)}, \alpha, \theta, \phi)$:

$$L(x^{(i)}, \alpha, \theta, \phi) = -D_{KL}(q_\phi(z|x^{(i)})||p_\alpha(z)) + \mathbb{E}_{q_\phi(z|x^{(i)})}[\log(p_\theta(x^{(i)}|z))]$$

The last term is approximated thanks to the unbiased Monte Carlo estimator, which provides us the following formula of the estimator of the ELBO $\hat{L}(x^{(i)}, \alpha, \theta, \phi)$:

$$\hat{L}(x^{(i)}, \alpha, \theta, \phi) = -D_{KL}(q_\phi(z|x^{(i)})||p_\alpha(z)) + \frac{1}{L}\sum_{i=1}^{L}[\log(p_\theta(x^{(i)}|z^{(i,l)}))]$$

Where $z^{(i,l)}$ is drawn from the target distribution $q_\phi(z|x^{(i)})$.

2.2 Standard VAE

The most commonly used VAE, as we will called it "Standard VAE", is defined as :

— Prior : $p(z) = \mathcal{N}(z; O, I_n)$;

— Likelihood : $p_\theta(x|z) = \mathcal{N}(x; \mu_\theta(z), \text{diag}(\sigma_\theta(z))^2)$;

— Target distribution : $q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \text{diag}(\sigma_\phi(x))^2)$.

A reparametrization trick is used here to make $q_\phi(z|x)$ differentiable :

$$q_\phi(z|x) = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon$$

Where ϵ is a multivariate Gaussian (with mean vector 0 and covariance matrix I) and $\odot$ is the point-wise operator.

We will use this standard VAE as a baseline to compare our models in our experiments.

3 Marginal tails of classic generative algorithms

3.1 Marginal tail of the Standard VAE

It has been observed that standard VAEs tend to produce light-tailed marginals, meaning that the probability of extreme values is lower than expected for a heavy-tailed distribution. This observation is consistent with previous results on generative adversarial networks (GANs) with normal priors and normalizing flows (NFs) with light-tailed base distributions.

The light-tailed marginals of standard VAEs can be attributed to the Lipschitz continuity of their neural network components. Neural networks composed of operations such as ReLUs, leaky ReLUs, linear layers, maxpooling, maxout activation, concatenation, or addition are piecewise linear operators with a finite number of linear regions. As a result, these neural networks are Lipschitz continuous with respect to Minkowski distances.

Proposition 6 *For the standard VAE with univariate output ($m = 1$), given that the neural network functions μ_θ and σ_θ of the probabilistic decoder are piecewise linear operators, then the output distribution sampled by the standard VAE is light-tailed.*

This proposition states that for a standard VAE with a univariate output, if the neural network functions μ_θ and σ_θ of the probabilistic decoder are piecewise linear operators, then the output distribution sampled by the standard VAE is light-tailed.

This proposition is based on the fact that μ_θ and σ_θ are Lipschitz continuous. Indeed, by computing the survival function of the variable X by integration over our latent variable Z, we have :

$$\mathbb{P}(X > u) = \int_z \mathbb{P}(X > u | Z = z) p(z) ds$$

The integral can be separated into two functions, the one on z such that $u - \mu_\theta(z) > 0$ and the other upon z such that $u - \mu_\theta(z) \leq 0$. The Lipschitz continuity of $\mu_\theta(z)$ enables us to show that the second function is bounded, and the Lipschitz continuity of $\sigma_\theta(z)$, combined with the dominated convergence theorem, shows that the first function $f_1(u)$ verifies for all $\alpha > 0$,

$$\lim_{u \to \infty} u^\alpha f_1(u) = 0$$

This result extends to multivariate outputs. The marginal distributions generated by the standard VAE are light-tailed whenever the neural networks functions μ_θ and σ_θ are composed of ReLUs, leaky ReLUs, linear layers, maxpooling, maxout, activation, concatenation or addition .

These findings have implications for the use of standard VAEs in applications that require heavy-tailed distributions, such as financial modeling or risk assessment. In these cases, it may be necessary to use alternative generative models, such as those based on normalizing flows with heavy-tailed base distributions.

3.2 Angular measure of generative algorithms

We consider here generative algorithms taking as an input a random vector and giving as an output a sample distribution.

Let's define the angular measure of a random vector $X \in (\mathbb{R}^+)^m$ as the following :

- We first decompose X into two components :
  - The radial component : $R = ||X||$;
  - The angular component : $\Theta = \frac{X}{||X||}$.
- If X has a a multivariate regular variation, the angular measure is defined as the probability measure S on the $(m-1)$-dimensional simplex such that :

$$P(\Theta \in \bullet | R > r) \xrightarrow{w} S(\bullet)$$

If the algorithm is a neural network composed of linear layers and ReLU activations, we can show that the angular measure of the sampled distribution is concentrated on a finite set of points.

It implies that, for an infinite radius, X is concentrated on specific axis. This result prevents us to generate correctly in the extreme regions. The chosen solution was to consider the polar decomposition of X as (R, Θ) and to enable the dependency between R and Θ by sampling Θ conditionally to the radius R.

4 VAE Architecture

In order to contravene the previously mentioned issues with the standard VAE which is unable to sample heavy-tailed distribution the authors suggested the following framework.

4.1 General framework

In order to generate a sample $x^{(i)}$ of a multivariate regularly varying random vector, we apply the following steps :

1. We draw the radius $r^{(i)}$ from a univariate heavy-tailed distribution R, using the VAE described in the subsection 4.2 ;

2. We sample $\Theta^{(i)}$ conditionally to $r^{(i)}$ (i.e. from the distribution $\Theta | R = r^{(i)}$), using the VAE described in the subsection 4.3 ;

3. We compute the point-wise operation : $X^{(i)} = r^{(i)} * \Theta^{(i)}$.

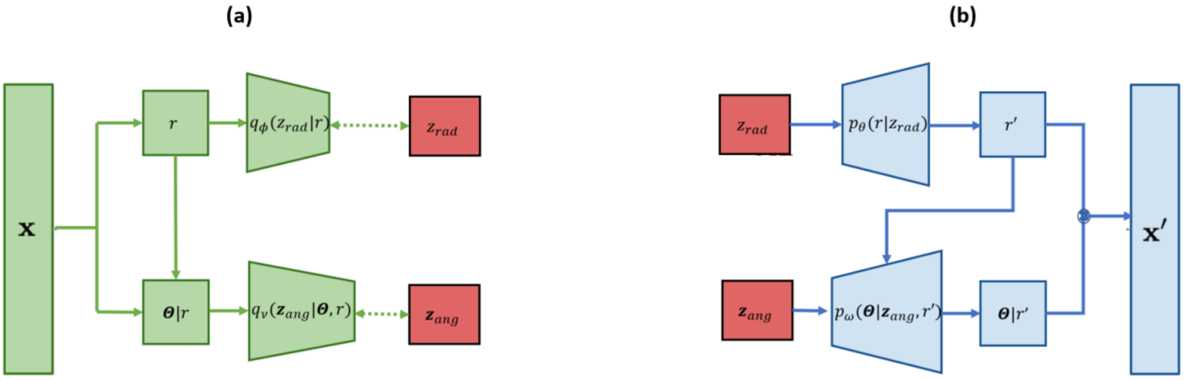


Figure 2: Global architecture of our approach with **(a)** the probabilistic encoders and **(b)** the probabilistic decoders. Ideally, distributions of $\mathbf{x}$ and $\mathbf{x}'$ are similar. Solid arrows show a causal link between the different blocks. Dashed double arrows in **(a)** indicate that the distributions in the pointed blocks are compared using a Kullback-Leibler divergence criterion (Equation 2).

4.2 VAE to sample the radial component

The latent component of the VAE to sample the radial component is denoted Z_{rad}. As Z_{rad} should follow a heavy-tailed distribution to respect the previous conditions, the authors chose that Z_{rad} follows the inverse-gamma distribution of strictly positive parameters α and β. They set $\beta = 1$ thanks to the proprieties of the inverse-gamma distribution (without loss of generality).

The framework to sample the radial component is defined as :

— Prior : $p_\alpha(z_{\text{rad}}) = f_{\text{Inv}\Gamma}(z_{\text{rad}}; \alpha, 1)$;

— Likelihood : $p_\theta(r|z_{\text{rad}}) = f_\Gamma(r; \alpha_\theta(z_{\text{rad}}), \beta_\theta(z_{\text{rad}}))$;

— Target : $q_\phi(z_{\text{rad}}|r) = f_{\text{Inv}\Gamma}(z_{\text{rad}}; \alpha_\phi(r), \beta_\phi(r))$.

Where f_Γ is the probability density function of the gamma distribution and $f_{\text{Inv}\Gamma}$ is the probability density function of the inverse gamma distribution. $\alpha_\theta, \beta_\theta, \alpha_\phi$ and β_ϕ are ReLU functions.

Proposition 12 *We consider the VAE parameterization described by the equations above. If we further assume that the function $\alpha_\theta(\cdot)$ is a strictly positive constant and the function $\beta_\theta(\cdot)$ satisfies :*

$$\lim_{z_{\text{rad}} \to \inf} \beta_\theta(z_{\text{rad}}) \propto \frac{1}{z_{\text{rad}}}$$

$$\lim_{z_{\text{rad}} \to 0} \beta_\theta(z_{\text{rad}}) \propto \frac{1}{z_{\text{rad}}}$$

then the univariate output distribution sampled by this VAE scheme is heavy-tailed with tail index equal to α.

The proof of this proposition is based on the fact that we can bound $z * \beta_\theta(z)$. It results that the function $f(r, z_{\text{rad}}) = p_\theta(r|z_{\text{rad}})p_\alpha(z_{\text{rad}})$ is bounded between two functions $f_1(r, z_{\text{rad}})$ and $f_2(r, z_{\text{rad}})$, both presenting asymptotically results enabling us to say that their integrals over $\mathbb{R}^+$ are proportional to $r^{-\alpha-1}$. As a result, $r^{\alpha+1}p(r)$ is bounded when $r \to \infty$ and $p(r)$ is therefore a heavy-tailed distribution with α as a tail index.

We conclude that the suggested VAE enables us to sample the radial component with a heavy-tailed distribution.

4.3 VAE to sample the angular component

As seen before, we use in this subsection a VAE adapted to sample the angular component Θ conditionally to the radius R. The chosen architecture was introduced in (2) to take into account the context of a dialog for dialog generation.

The latent variable of our conditional VAE is called Z_{ang} and we make the same assumption as in (2) to consider that Z_{ang} follows a multivariate Gaussian distribution. The article (2) also assumes that the target distribution $q_\omega(z_{\text{ang}}|\Theta, r)$ follows a Gaussian distribution.

The conditional angular VAE is defined as below :

— $p(z_{\text{ang}}) = \mathcal{N}(z_{\text{ang}}; 0, I_n)$;

— $p_\nu(\Theta|z_{\text{ang}}, r) = \int_{S(\Theta)} \mathcal{N}(s; \mu_\nu(z_{\text{ang}}, r), diag(\sigma_\nu(z_{\text{ang}}, r))^2)$;

— $q_\omega(z_{\text{ang}}|\Theta, r) = \mathcal{N}(z_{\text{ang}}; \mu_\omega(\Theta, r), diag(\sigma_\omega(\Theta, r))^2)$.

The authors defined $S(\Theta)$ as the $\mathcal{L}_1$-sphere. In practice, the likelihood is in fact sampled on a Dirichlet distribution, in order to sample on the multivariate $(m-1)$-simplex.

5 Implementation

In this section, we elaborate on the specific architectural configurations utilized in the trained Variational Autoencoders (VAEs) during our numerical experiments.

5.1 Radius Generation VAE

For the probabilistic encoder, we adopt two 5-dimensional hidden layers with Rectified Linear Unit (ReLU) activation functions. The output layer comprises a 2-dimensional dense layer with ReLU activation. To expedite convergence, we initialize the weights of the dense layers to 0 and set their biases to strictly positive values sampled from a uniform distribution between 1 and 2.

Regarding the probabilistic decoder, composed of f_θ and g_θ, defined as :

$$\beta_\theta(z_{\text{rad}}) = \frac{|f_\theta(z_{\text{rad}})|}{z_{\text{rad}}^2}$$

an

$$\alpha_\theta(z_{\text{rad}}) = \frac{|g_\theta(z_{\text{rad}})|}{z_{\text{rad}}}$$

It mirrors the architecture of the probabilistic encoder.
However, for the output layers, one corresponds to the output f_θ, initialized with a strictly positive output bias (randomly sampled from a uniform distribution between 1 and 2), and the other corresponds to the output of g_θ, initialized with a positive output kernel (randomly sampled from a uniform distribution between 0.1 and 2).

5.2 Angular VAE

The encoder involves a latent dimension of 4 and comprises 3 hidden layers with ReLU activation functions—each layer yields 8, 8, and 4 output features, respectively. The output layer is a dense linear layer. Standard initialization methods are employed for the encoder.

For the decoder, the input radius undergoes transformation according to the following equations :

$$\mu_\nu(z_{\text{ang}}, r) = f_\nu\left(z_{\text{ang}}, \frac{1}{1+r}\right)$$

and

$$\sigma_\nu(z_{\text{ang}}, r) = g_\nu\left(z_{\text{ang}}, \frac{1}{1+r}\right)$$

Where f_ν and g_{nu} are Lipschitz continuous neural networks.

The decoder includes 3 hidden layers with ReLU activation—resulting in 5, 10, and 5 output features, respectively. The output layer is a dense layer. Standard initialization is used for the decoder, except for the bias of the final layer, which is initialized by sampling from a uniform distribution between 0.5 and 3.

5.3 Learning Set-up

The training procedure for our hierarchical architecture involves two distinct training losses : L_R for the radius VAE and $L_{\Theta|R}$ for the angular VAE. We derive L_R from the formula of the estimator of the ELBO given in section 2 applied to the prior and the target distributions described in 4.2, and $L_{\Theta|R}$ from the respective equations in the text.

The overall training loss L_{ExtVAE} is then the sum of L_R and $L_{\Theta|R}$. We first train the radius VAE parameters (α, θ, ϕ) and subsequently the angular VAE parameters (ν, ω). Depending on experiments, α can be known or unknown. When known, it is set to the desired value in the calculus of the overall loss ; otherwise, it is optimized via gradient descent.

For estimating (α, θ, ϕ), the training duration is limited to 5000 epochs with a learning rate of 10^{-4}. Similarly, for estimating (ν, ω), the training duration spans 5000 epochs, but the learning rate is fixed at 10^{-5}. Both cases employ the Adam optimizer with a batch size of 32. The implementation predominantly leverages the TensorFlow and TensorFlow-Probability libraries, and the complete codebase is publicly accessible.

5.4 Datasets

5.5 Synthesized Data Sets

We sample in a space of dimension 5. We consider a sampling setting for the radius distribution denoted R1 such that

$$R_1 \sim 2U \times Inv\Gamma(\alpha_1 = 1.5; \beta = 0.6)$$

with U uniform on [0, 1]. From Breiman's Lemma, the radius distribution is heavy-tailed with tail index α_1. The detailed expression of the conditional angular distribution $\theta_1|R_1 = r$ is given by

$$\theta_1|R_1 = r \sim Diri(\alpha_1(r), \alpha_1(r), \alpha_2(r), \alpha_2(r), \alpha_2(r))$$

where $\alpha_1(r) = 3(2 - min(1, 1/2r)), \alpha_2(r) = 3(1 + min(1, 1/2r))$

5.6 Danube River Network Discharge Measurements

The upper Danube basin is an European river network which drainage basin covers a large part of Austria, Switzerland and of the south of Germany. Danube river network data set is available from the Bavarian Environmental Agency at http ://www.gkd.bayern.de. As river discharges usually exhibit heavy-tailed distribution, this data set have been extensively studied in the community of multivariate extremes0

We consider measurements from a subset of 5 stations from which we want to sample.

6 Experiments

Our experiments have shown results similar to those in the article and confirm the theoretical parts of the study. In order to visualize the advantages of the proposed method, we compare the StandardVAE with both our implementations, ExtVAE when the tail index is known and UExtVAE when the tail index in unknown.

6.1 Implementations

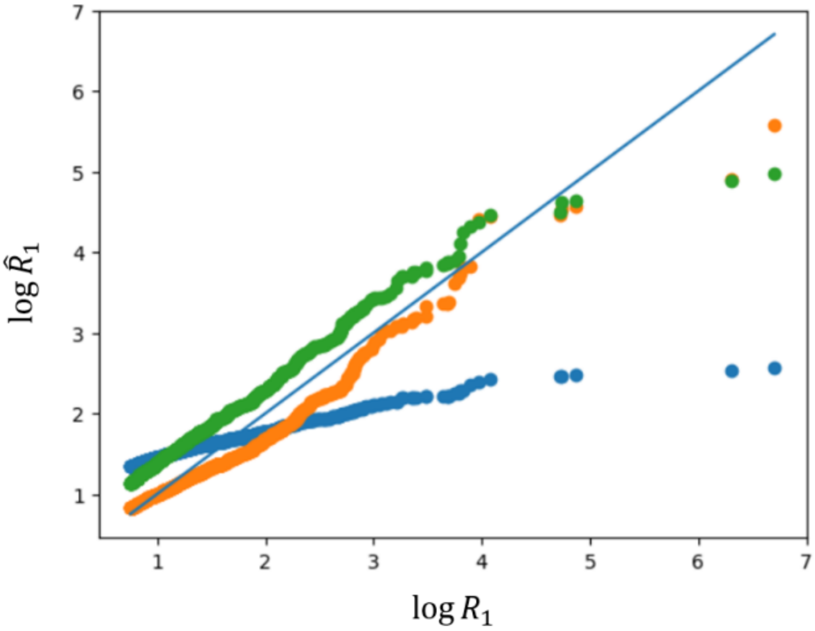
In our implementation, we pursued two distinct approaches for building the variational autoencoders (VAEs).

The first version utilized TensorFlow, leveraging its predefined distributions and functions, simplifying the process through the use of established tools. We employed TensorFlow Probability's modules like DistributionLambda to create distributions and KLDivergenceRegularizer for regularization, constructing the VAE model with predefined functionalities.

However, in our quest for stability and efficiency, we opted for a second approach using PyTorch. In this manual implementation, we crafted the necessary tools from scratch, such as the IndependentNormal class, where we designed the mean and scale parameters manually within a PyTorch module. Despite the deeper control and flexibility PyTorch offered, enabling us to define models and their components at a granular level, we faced challenges in achieving stability and robustness for the conditional VAE specifically tailored for temporal angular data.

As a result, we conducted experiments and found that the simpler TensorFlow version provided more stability and met our requirements for the given angular temporal data, leading us to adopt this TensorFlow-based approach for our final implementation.

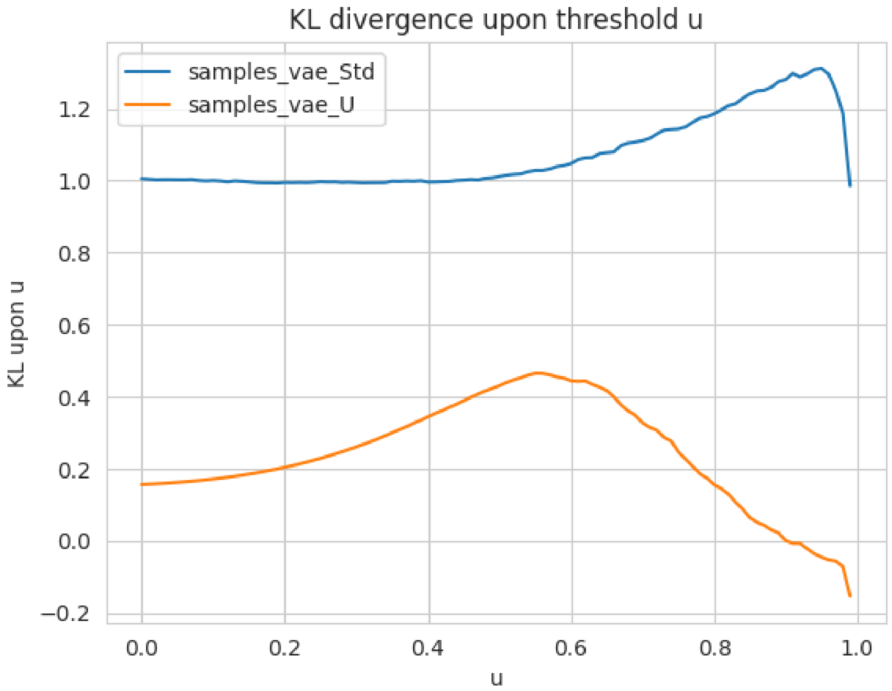
6.2 Quantile-Quantile plot



Ploting the log-QQ plot of the three VAEs shows the contribution of the proposed method.

In particular, by plotting the upper decile of 10 000 radii samples from the different VAEs and the true upper decile of our test data, we find that the our implementations of ExtVAE and UExtVAE better follows the ideal line of the radius distribution, whereas StandardVAE deviates.

6.3 Kullback-Leibler divergence



Moreover, plotting the Kullback-Leibler divergence of each VAE confirms that our models achieve better results than the StandardVAE. The latter quickly reach a high value of the Kullback-Leibler divergence, while both our implementations present low values of this divergence. We also note that the divergences are very close for UExtVAE, suggesting that the estimation of the tail index is efficient in the case of the UExtVAE.

6.4 Hill Estimator

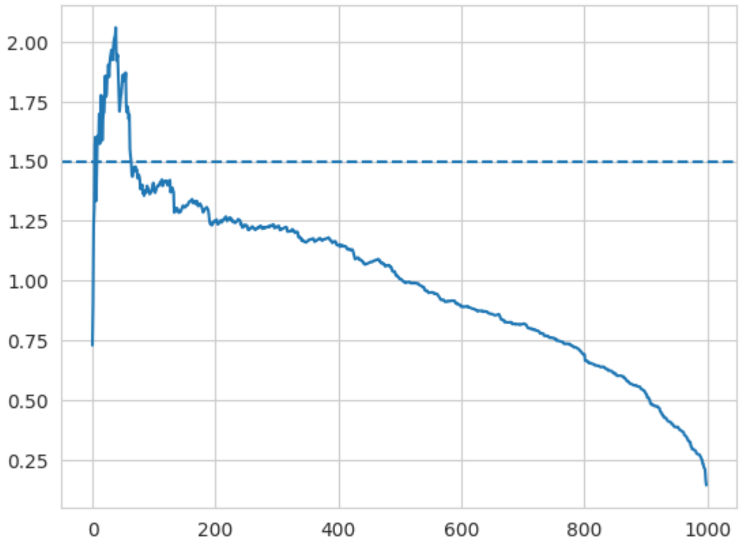


FIGURE 2 – Hill Estimator

Estimating the tail index of a univariate distribution from samples is challenging. The Hill plot, a common tool in extreme value analysis, aims to estimate this index. However, its effectiveness relies on observing a plateau in the plot from certain order statistics, indicating a consistent estimator for the tail index. In some cases, this method may not provide a clear plateau, limiting its usefulness

6.5 Angular measures

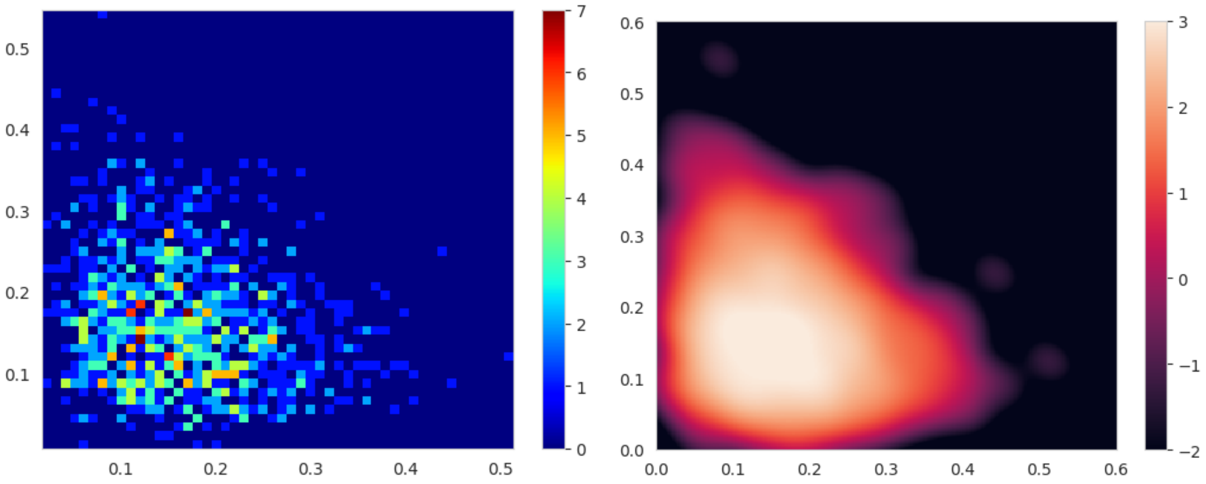


FIGURE 3 – Log Probabilities of the true distributions

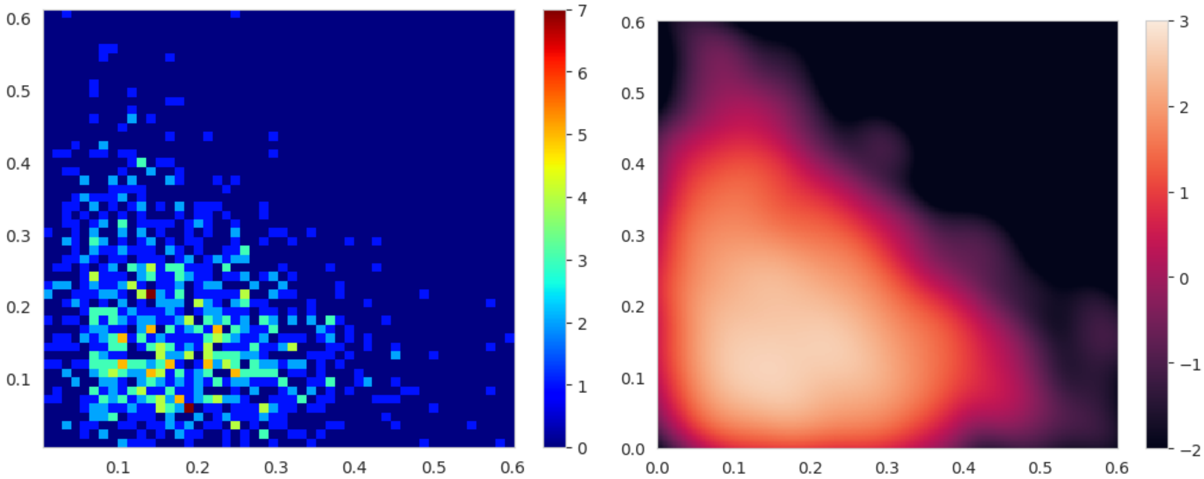


Figure 4 – Log Probabilities of the obtained distributions

To visualize the resulting angular distribution, we decided, as in the article, to compare the log probabilities of the angular measures between the true distribution and the distribution provided by the ExtVAE. We observe that the distributions are quite similar. In particular, our ExtVAE provides a large range of values, which suggests that our implementation does not generate on a restricted number of axis. The proposed framework decomposed X as (R,Θ) in order to avoid the concentration of X on specific axis, and this experiment shows that the solution actually works. .

7 Conclusion

In this article, the gap between machine learning and Extreme Values Theory was narrowed thanks to the adaptation of a Variational AutoEncoder to sample a multivariate heavy-tail distribution, oftenly used in extreme values theory in a large range of applications.

The framework proposed to separate the sampled variable into its radial and its angular components. The radius is generated thanks to an adapted VAE to sample a univariate heavy-tailed distribution. The angle is then generated through a VAE conditional to the previously-generated radius, in order to capture the dependency between both components.

The experiments show the efficiency of this approach and the concordance between the theoretical results and the practical ones in a real-case scenario. The article therefore pushes forward methods to efficiently generate multivariate heavy-tailed distributions.

Références

[1] N. Lafon, P. Naveau, and R. Fablet, "A vae approach to sample multivariate extremes," 2023.

[2] T. Zhao, R. Zhao, and M. Eskenazi, "Learning discourse-level diversity for neural dialog models using conditional variational autoencoders," 2017.