Quasi-Monte Carlo for Efficient Fourier Pricing of Multi-Asset Options: Implementation in Premia

Michael Samet*[1] and Chiheb Ben Hammouda[2]

[1]Chair of Mathematics for Uncertainty Quantification, RWTH Aachen University, Aachen, Germany.
[2]Mathematical Institute, Utrecht University, Utrecht, the Netherlands

Abstract

This work advocates using the randomized quasi-Monte Carlo (RQMC) quadrature to improve the scalability of Fourier methods for pricing multi-asset options. The RQMC technique benefits from the smoothness of the integrand and alleviates the curse of dimensionality while providing practical error estimates. Nonetheless, the applicability of RQMC on the unbounded domain, $\mathbb{R}^d$, requires a domain transformation to $[0,1]^d$, which may result in singularities of the transformed integrand at the corners of the hypercube, and deteriorate the rate of convergence of RQMC. To circumvent this difficulty, we design an efficient domain transformation procedure based on the derived boundary growth conditions of the integrand. We demonstrate the efficiency of employing RQMC with an appropriate transformation to price options in the Fourier space for various pricing models, payoffs, and dimensions.

1 Fourier Valuation Formula

The aim is to efficiently compute the price of multi-asset European option when the characteristic function is known at least semi-analytically. To accomplish this aim, we use the multivariate representation of the option price in the Fourier domain as in Bayer et al. [2023].

Notation 1.1 (Notations and Definitions)**.**

- $\boldsymbol{X}_t := \left(X_t^1, \ldots, X_t^d\right)$ is a d-dimensional ($d \in \mathbb{N}$) vector of log-asset prices[1] whose dynamics follow a multivariate stochastic model with the market parameters denoted by the vector $\boldsymbol{\Theta}_X$.
- $\Phi_{\boldsymbol{X}_T}(\boldsymbol{z}) := \mathbb{E}[e^{\mathrm{i}\boldsymbol{z}^\top \boldsymbol{X}_T}]$, for $\boldsymbol{z} \in \mathbb{C}^d$, denotes the extended characteristic function, where $\boldsymbol{z}^\top$ represents the transpose of $\boldsymbol{z}$. We define the bilinear form $\boldsymbol{x}^\top \boldsymbol{y} = \sum_{j=1}^d x_j y_j$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^d$.
- $P : \mathbb{R}^d \mapsto \mathbb{R}_+$ denotes the payoff function, and $\widehat{P}(\boldsymbol{z}) := \int_{\mathbb{R}^d} e^{\mathrm{i}\boldsymbol{z}^\top \boldsymbol{x}} P(\boldsymbol{x}) d\boldsymbol{x}$, for $\boldsymbol{z} \in \mathbb{C}^d$ represents its extended Fourier transform.

*samet@uq.rwth-aachen.de

[1] $X_t^i := \log(S_t^i), i = 1, \ldots, d$, $\{S_t^i\}_{i=1}^d$ are asset prices at time $t > 0$.

- $\boldsymbol{\Theta}_P = (K, T, r, q)$ represents the vector of the payoff parameters, where K denotes the strike price, T is the maturity time, $r \geq 0$ is a constant risk-free interest rate, and $q \geq 0$ is a constant dividend yield rate.
- i denotes the unit imaginary number, and $\Re[\cdot]$ and $\Im[\cdot]$ represent the real and imaginary parts of a complex number, respectively.
- $\Gamma(z)$ is the complex Gamma function defined for $z \in \mathbb{C}$ with $\Re[z] > 0$. The numerical computation of the Gamma function is done by the Lanczos approximation Lanczos [1964].
- $K_\lambda(y)$ is the modified Bessel function of the second kind defined for $y \in \mathbb{R}_+$.

Assumption 1.2 (Assumptions on the payoff)**.**

- $\delta_P := \{\boldsymbol{R} \in \mathbb{R}^d \mid \boldsymbol{x} \mapsto e^{\boldsymbol{R}^\top \boldsymbol{x}} P(\boldsymbol{x}) \in L^1(\mathbb{R}^d)\} \neq \emptyset$: strip of analyticity of the payoff Fourier transform.

Assumption 1.3 (Assumptions on the model)**.**

- $\delta_X := \{\boldsymbol{R} \in \mathbb{R}^d \mid \boldsymbol{y} \mapsto \mid \Phi_{\boldsymbol{X}_T}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R}) \mid < \infty, \text{and } \boldsymbol{y} \mapsto \Phi_{\boldsymbol{X}_T}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R}) \in L^1(\mathbb{R}^d)\} \neq \emptyset$: strip of analyticity of the extended characteristic function.

Proposition 1.4 (Multivariate Fourier Pricing Valuation Formula)**.** *We employ the notation in 1.1 and suppose that Assumptions 1.2 and 1.3 hold and that $\delta_V = \delta_X \cap \delta_P$. Then, for $\boldsymbol{R} \in \delta_V$, the option value is given by the following:*

$$V(\boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P) = (2\pi)^{-d} e^{-rT} \int_{\mathbb{R}^d} \Re\left[\Phi_{\boldsymbol{X}_T}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R}) \widehat{P}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R})\right] d\boldsymbol{y}. \tag{1.1}$$

Proof. For the proof of Proposition 1.4, refer to Bayer et al. [2023]. □

From (1.1), we define the integrand of interest as follows:

$$g(\boldsymbol{y}; \boldsymbol{R}, \boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P) := (2\pi)^{-d} e^{-rT} \Re[\Phi_{\boldsymbol{X}_T}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R}) \widehat{P}(\boldsymbol{y} + \mathrm{i}\boldsymbol{R})], \boldsymbol{y} \in \mathbb{R}^d, \boldsymbol{R} \in \delta_V. \tag{1.2}$$

Bayer et al. [2023] proposed a rule for the choice of the damping parameters, $\boldsymbol{R}$, that leads to a regular integrand, and numerical evidence shows that their rule can accelerate the convergence of numerical quadrature methods significantly. Their proposed rule is given by:

$$\mathbf{R}^* = \underset{\mathbf{R} \in \delta_V}{\arg\min}\, g(\mathbf{0}; \mathbf{R}, \boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P). \tag{1.3}$$

For better numerical stability, we recommend optimizing the logarithm of the integrand

$$\mathbf{R}^* = \underset{\mathbf{R} \in \delta_V}{\arg\min}\, \log(g(\mathbf{0}; \mathbf{R}, \boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P)). \tag{1.4}$$

Implementation details related to the optimization of the damping parameters are located in the file `damping_optimization.py`. More specifically, the method `optimize_damping_parameters` defines the constraints of the optimization problem (δ_V), where `log_integrand_to_optimize` defines the objective function of the optimization problem ($\log(g(\mathbf{0}; \mathbf{R}, \boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P))$. The constrained optimization problem in (1.4) can be effectively solved using, for instance, a constrained trust-region algorithm provided by the SciPy library.

2 Numerical Method - Randomized Quasi-Monte Carlo (RQMC)

This section introduces the RQMC method. The QMC estimator of an integral of a function, $f : [0,1]^d \mapsto \mathbb{R}$, is an N-point equal-weighted quadrature rule denoted by $Q_{N,d}^{QMC}[\cdot]$, and reads as Dick et al. [2013]:

$$I[f] := \int_{[0,1]^d} f(\boldsymbol{u}) d\boldsymbol{u} \approx Q_{N,d}^{QMC}[f] := \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{u}_n), \tag{2.1}$$

where $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ is a set of deterministic low discrepancy (LD) sequences (e.g., Sobol, Niederreiter, Halton, Hammersley, and Faure, see Dick et al. [2013]), $\boldsymbol{u}_n \in [0,1]^d, n \in \{1, \ldots, N\}$. The randomized variant of the QMC estimator (2.1), the RQMC estimator, is given by:

$$Q_{N,S,d}^{RQMC}[f] := \frac{1}{S} \sum_{s=1}^{S} \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{u}_n^{(s)}), \tag{2.2}$$

where $\{\boldsymbol{u}_n\}_{n=1}^N$ is the sequence of deterministic QMC points, and for $n = 1, \ldots N$, $\{\boldsymbol{u}_n^{(s)}\}_{s=1}^S$ is obtained by the appropriate randomization of $\{\boldsymbol{u}_n\}_{n=1}^N$, such that $\boldsymbol{u}_n^{(s)} \sim \mathcal{U}([0,1]^d)$. For fixed $n = 1, \ldots, N$, $\boldsymbol{u}_n^{(s)}$ are independent for any $s = 1, \ldots, S$. The set of points $\{\boldsymbol{u}_n^{(s)}\}_{s=1}^S$ yields an unbiased estimator (2.2) (i.e., $\mathbb{E}[Q_{N,S,d}^{RQMC}[f]] = I[f]$). This default choice of QMC points in this implementation is the Sobol sequences Sobol' et al. [2011] with digital shifting for the randomization as proposed by Cranley and Patterson [1976]. For the sampling of QMC points, we use the Python open source library QMCPy Choi et al. [2020]. The randomization of the LD points enables the derivation of the root mean squared error of the estimator, given by Dick et al. [2013]

$$\mathcal{E}_{N,S,d}^{RQMC}[f] := \frac{C_\alpha}{\sqrt{S}} \sqrt{\frac{1}{S-1} \sum_{s=1}^{S} \left(\frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{u}_n^{(s)}) - Q_{N,S,d}^{RQMC}[f] \right)^2}, \tag{2.3}$$

where C_α denotes the $(1 - \frac{\alpha}{2})$-quantile of the standard normal distribution for a confidence level $0 < \alpha \ll 1$. In this work, we set by default $C_\alpha = 1.96$ which corresponds to $\alpha = 0.05$. Moreover, (2.3) reveals that the statistical error can be controlled by the number of shifts, S, which we set by default to $S = 30$ as a rule of thumb, and the user can change the number of randomizations. However, we want to highlight that the error estimation in RQMC is based on the central limit theorem, and hence it is an asymptotic result that holds for $S \to \infty$ L'Ecuyer [2018]. Consequently, smaller values of S can lead to inaccurate error estimates. The main QMC constructions are restricted to the generation of LD point sets on $[0,1]^d$ Dick et al. [2013]. Consequently, an appropriate integral transformation is necessary to apply RQMC to the unbounded domain, $\mathbb{R}^d$. The following mapping is the typical transformation in the literature to map an integrand from $\mathbb{R}^d$ to $[0,1]^d$:

$$\boldsymbol{u} = \Psi(\mathbf{y}; \boldsymbol{\Lambda}) \Leftrightarrow \mathbf{y} = \Psi^{-1}(\boldsymbol{u}; \boldsymbol{\Lambda}), \boldsymbol{u} \in [0,1]^d, \tag{2.4}$$

where $\boldsymbol{y} \mapsto \Psi(\boldsymbol{y}; \boldsymbol{\Lambda})$ is the cumulative distribution function (CDF) of some $\mathbb{R}^d$-valued RV with parameter vector $\boldsymbol{\Lambda}$ (e.g., mean, scale, and degrees of freedom), and $\Psi^{-1}(\boldsymbol{u}; \boldsymbol{\Lambda})$ is the inverse CDF (ICDF). By applying the mapping 2.4 to the integral in (1.1), we obtain the following:

$$
\begin{aligned}
V\left(\boldsymbol{\Theta}_X, \boldsymbol{\Theta}_P\right) &= \int_{[0,1]^d} \tilde{g}(\boldsymbol{u}) d\boldsymbol{u} \\
\tilde{g}(\boldsymbol{u}) &:= \frac{g \circ \Psi^{-1}(\boldsymbol{u})}{\psi \circ \Psi^{-1}(\boldsymbol{u})}, \quad \boldsymbol{u} \in [0,1]^d
\end{aligned} \tag{2.5}
$$

where $g(\cdot)$ is the integrand defined in 1.2, and $\psi(\cdot)$ denotes the probability density function (PDF) supported on $\mathbb{R}^d$, corresponding to $\Psi(\cdot)$. In Section 5, we discuss the choice of ψ and its parameters, for more details on the associated challenges we refer to Bayer et al. [2024]

The transformed Fourier integrand in our framework reads as follows, for $\boldsymbol{u} \in [0,1]^d$ and $\boldsymbol{R} \in \delta_V$

$$
\tilde{g}(\boldsymbol{u}) = \frac{e^{-rT} e^{-\boldsymbol{R}^\top \boldsymbol{X}_0}}{(2\pi)^d} \Re \left[e^{\mathrm{i}\left(\Psi^{-1}(\boldsymbol{u})\right)^\top \boldsymbol{X}_0} \widehat{P}\left(\Psi^{-1}(\boldsymbol{u}) + \mathrm{i}\boldsymbol{R}\right) \frac{\Phi_{\boldsymbol{X}_T}\left(\Psi^{-1}(\boldsymbol{u}) + \mathrm{i}\boldsymbol{R}\right)}{\psi\left(\Psi^{-1}(\boldsymbol{u})\right)} \right].
$$

3 Pricing Models and their Characteristic Functions

This Section describes the considered asset price models and their corresponding characteristic functions. The details related to characteristic functions for each model are located in the file `models.py`. Each model includes the following modules: `characteristic_function` ($\Phi_{\boldsymbol{X}_T}(\cdot)$), `strip_of_analyticity` ($\delta_X$), and `compute_mu` ($\boldsymbol{\mu}$).

Example 3.1 (Geometric Brownian Motion (GBM))**.** The discounted characteristic function of the GBM model under the risk-neutral pricing measure is given in the form of $\Phi^{\mathrm{GBM}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = e^{\mathrm{i}\boldsymbol{z}^\top(\boldsymbol{X}_0 + (r-q+\boldsymbol{\mu}_{\mathrm{GBM}})T)} \phi^{\mathrm{GBM}}_{\boldsymbol{X}_T}(\boldsymbol{z})$ for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta^{\mathrm{GBM}}_X = \mathbb{R}^d$, by Oosterlee and Grzelak [2019]

$$
\phi^{\mathrm{GBM}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = \exp\left(-\frac{T}{2}\boldsymbol{z}^\top \boldsymbol{\Sigma} \boldsymbol{z}\right), \quad \Im[\boldsymbol{z}] \in \delta^{\mathrm{GBM}}_X. \tag{3.1}
$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_d) \in \mathbb{R}^d_+$ is the vector of volatilities, and we denote by $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$ the covariance matrix of the log returns i.e. $\boldsymbol{\Sigma}_{ij} = \rho_{i,j}\sigma_i\sigma_r j$, with $\rho_{i,j}$ denoting the correlation between the Brownian motions of the i^{th} and j^{th} asset price processes. Moreover, $\boldsymbol{\mu}_{\mathrm{GBM}}$ is a vector of drift correction terms that ensure that $\left\{e^{-(r-q)t} S^j_t\right\}_{t \geq 0}$ is a martingale for all $j = 1, \ldots d$, and is given by

$$
\mu^j_{\mathrm{GBM}} = -\frac{\sigma_j^2}{2}, \quad j = 1, \ldots, d.
$$

Example 3.2 (Variance Gamma (VG))**.** The discounted characteristic function of the VG model under the risk-neutral pricing measure is given in the form of $\Phi^{\mathrm{VG}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = e^{\mathrm{i}\boldsymbol{z}^\top(\boldsymbol{X}_0 + (r-q+\boldsymbol{\mu}_{\mathrm{VG}})T)} \phi^{\mathrm{VG}}_{\boldsymbol{X}_T}(\boldsymbol{z})$ for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta^{\mathrm{VG}}_X = \left\{\boldsymbol{R} \in \mathbb{R}^d \,\middle|\, 1 + \nu \boldsymbol{R}^\top \boldsymbol{\theta} - \frac{1}{2}\nu \boldsymbol{R}^\top \boldsymbol{\Sigma} \boldsymbol{R} > 0\right\}$, by Luciano and Schoutens [2006]

$$
\phi^{\mathrm{VG}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = \left(1 - \mathrm{i}\nu \boldsymbol{z}^\top \boldsymbol{\theta} + \frac{1}{2}\nu \boldsymbol{z}^\top \boldsymbol{\Sigma} \boldsymbol{z}\right)^{-T/\nu}, \quad \Im[\boldsymbol{z}] \in \delta^{\mathrm{VG}}_X. \tag{3.2}
$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_d) \in \mathbb{R}^d_+$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d) \in \mathbb{R}^d$, $\nu > 0$, and $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ denotes the covariance matrix of the log returns i.e. $\boldsymbol{\Sigma}_{ij} = \rho_{i,j}\sigma_i\sigma_j$ with $\rho_{i,j}$ denoting the correlation between the Brownian motions of the i^{th} and j^{th} asset price processes. Moreover, $\boldsymbol{\mu}_{VG}$ is a vector drift correction terms that ensures that $\left\{e^{-(r-q)t}S^j_t \mid t \geq 0\right\}$ is a martingale for all $j = 1, \ldots d$, and is given by

$$\mu^j_{VG} = \frac{1}{\nu}\log\left(1 - \frac{1}{2}\sigma_j^2\nu - \theta_j\nu\right), \quad j = 1, \ldots, d.$$

Example 3.3 (Normal Inverse Gaussian (NIG))**.** The discounted characteristic function of the NIG model under the risk-neutral pricing measure is given in the form of $\Phi^{\text{NIG}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = e^{\mathrm{i}\boldsymbol{z}^\top(\boldsymbol{X}_0 + (r-q+\boldsymbol{\mu}_{\text{NIG}})T)}\phi^{\text{NIG}}_{\boldsymbol{X}_T}(\boldsymbol{z})$ for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta^{\text{NIG}}_X = \left\{\boldsymbol{R} \in \mathbb{R}^d \mid \alpha^2 - (\boldsymbol{\beta} - \boldsymbol{R})^\top\boldsymbol{\Delta}(\boldsymbol{\beta} - \boldsymbol{R}) > 0\right\}$, by Prause et al. [1999]

$$\phi^{\text{NIG}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = \exp\left(\delta T\left(\sqrt{\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}} - \sqrt{\alpha^2 - (\boldsymbol{\beta} + \mathrm{i}\boldsymbol{z})^\top\boldsymbol{\Delta}(\boldsymbol{\beta} + \mathrm{i}\boldsymbol{z})}\right)\right), \quad \Im[\boldsymbol{z}] \in \delta^{\text{NIG}}_X. \tag{3.3}$$

where $\alpha \in \mathbb{R}_+, \delta \in \mathbb{R}_+, \boldsymbol{\beta} \in \mathbb{R}^d$ with $\alpha^2 > \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\Delta}\boldsymbol{\beta}$, and $\boldsymbol{\Delta} \in \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix with a unit determinant i.e. $|\det(\boldsymbol{\Delta})| = 1$, related to the covariance matrix of the log returns as follows Eberlein et al. [2010]

$$\boldsymbol{\Sigma} = \delta\left(\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}\right)^{-\frac{1}{2}}\left(\boldsymbol{\Delta} + \left(\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}\right)^{-1}\boldsymbol{\Delta}\boldsymbol{\beta}\boldsymbol{\beta}^\top\boldsymbol{\Delta}\right)$$

Moreover, $\boldsymbol{\mu}_{\text{NIG}}$ is a vector drift correction terms that ensures that $\left\{e^{-(r-q)t}S^j_t \mid t \geq 0\right\}$ is a martingale for all $j = 1, \ldots d$, and is given by

$$\mu^j_{\text{NIG}} = -\delta\left(\sqrt{\alpha^2 - \beta_j^2} - \sqrt{\alpha^2 - (\beta_j + 1)^2}\right), \quad j = 1, \ldots, d.$$

Example 3.4 (Generalized Hyperbolic (GH))**.** The discounted characteristic function of the GH model under the risk-neutral pricing measure is given in the form of $\Phi^{\text{GH}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = e^{\mathrm{i}\boldsymbol{z}^\top(\boldsymbol{X}_0 + (r-q+\boldsymbol{\mu}_{\text{GH}})T)}\phi^{\text{GH}}_{\boldsymbol{X}_T}(\boldsymbol{z})$ for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R}$, with $\boldsymbol{R} \in \delta^{\text{GH}}_X = \left\{\boldsymbol{R} \in \mathbb{R}^d \mid \alpha^2 - (\boldsymbol{\beta} - \boldsymbol{R})^\top\boldsymbol{\Delta}(\boldsymbol{\beta} - \boldsymbol{R}) > 0\right\}$, by Prause et al. [1999]

$$\phi^{\text{GH}}_{\boldsymbol{X}_T}(\boldsymbol{z}) = \left(\frac{\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}}{\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta} + \boldsymbol{z}^\top\boldsymbol{\Delta}\boldsymbol{z} - 2i\boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{z}}\right)^{\lambda/2}\frac{\mathrm{K}_\lambda\left(\delta T\sqrt{\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta} + \boldsymbol{z}^\top\boldsymbol{\Delta}\boldsymbol{z} - 2i\boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{z}}\right)}{\mathrm{K}_\lambda\left(\delta T\sqrt{\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}}\right)}, \quad \Im[\boldsymbol{z}] \in \delta^{\text{GH}}_X. \tag{3.4}$$

where $\alpha \in \mathbb{R}_+, \delta \in \mathbb{R}_+, \boldsymbol{\beta} \in \mathbb{R}^d, \lambda \in \mathbb{R}$ with $\alpha^2 > \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\Delta}\boldsymbol{\beta}$, where $\boldsymbol{\Delta} \in \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix with a unit determinant i.e. $|\det(\boldsymbol{\Delta})| = 1$, related to the covariance matrix of the log returns as follows Eberlein et al. [2010]

$$\boldsymbol{\Sigma} = \delta\left(\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}\right)^{-\frac{1}{2}}\left(\boldsymbol{\Delta} + \left(\alpha^2 - \boldsymbol{\beta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}\right)^{-1}\boldsymbol{\Delta}\boldsymbol{\beta}\boldsymbol{\beta}^\top\boldsymbol{\Delta}\right)$$

Moreover, is a vector drift correction terms that ensures that $\left\{e^{-(r-q)t}S^j_t \mid t \geq 0\right\}$ is a martingale for all $j = 1, \ldots d$. In the case of the GH model, we do not have an explicit expression for $\boldsymbol{\mu}_{\text{GH}}$,

hence we compute it by evaluating the 1D characteristic function as follows

$$\mu^j_{\mathrm{GH}} = -\frac{1}{T}\log\left(\phi^{\mathrm{GH}}_{X^j_T}(-\mathrm{i})\right), \quad j = 1, \ldots, d$$

where

$$\phi^{\mathrm{GH}}_{X^j_T}(z) = \left(\frac{\alpha^2-\beta^2}{\alpha^2-(\beta+iz)^2}\right)^{\lambda/2} \frac{\mathrm{K}_\lambda\left(\delta\sqrt{\alpha^2-(\beta+iz)^2}\right)}{\mathrm{K}_\lambda\left(\delta\sqrt{\alpha^2-\beta^2}\right)}, \quad \Im[z] \in \delta^{\mathrm{GH}}_X. \tag{3.5}$$

Remark 3.5. The GH model coincides with the NIG model for $\lambda = -\frac{1}{2}$ Prause et al. [1999].

4 Payoff Functions and their Fourier Tranforms

This Section describes the considered option payoffs and their Fourier transforms. The details related to each of the considered payoff functions are located in the file `payoffs.py`. Each payoff includes the following modules: `fourier_transform` ($\widehat{P}(\cdot)$), `strip_of_analyticity` (δ_P), `calculate_X0` (definition of $\boldsymbol{X}_0$) `scaling` (scaling factor of the integrand for scaled payoffs).

Example 4.1 (Basket Put)**.** The payoff function of the basket put option on d stocks is given by:

$$P(\boldsymbol{X}_T) = \max\left(K - \sum_{j=1}^{d} w_j e^{X^j_T}, 0\right), \quad w_j > 0,\ \sum_{j=1}^{d} w_j = 1. \tag{4.1}$$

If the weights of the basket are not manually specified by the user, the default value is $w_j = \frac{1}{d}$ for $j = 1, \ldots, d$. The payoff function can be straightforwardly scaled to become independent of the strike and weights values by defining $X^j_t = \log\left(\frac{w_j S^j_t}{K}\right)$. The Fourier transform of the scaled basket put option is defined for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta_P = \{\boldsymbol{R} \in \mathbb{R}^d \mid R_j > 0, j = 1, \ldots, d\}$, by Hurd and Zhou [2010]

$$\widehat{P}(\boldsymbol{z}) = \frac{\prod_{j=1}^{d}\Gamma(-\mathrm{i}z_j)}{\Gamma\left(-\mathrm{i}\sum_{j=1}^{d} z_j + 2\right)}. \tag{4.2}$$

Example 4.2 (Spread Call)**.** The payoff function of the spread call option on d stocks is given by:

$$P(\boldsymbol{X}_T) = \max\left(e^{X^1_T} - \sum_{j=2}^{d} e^{X^j_T} - K, 0\right). \tag{4.3}$$

The payoff function can be straightforwardly scaled to become independent of the strike by defining $X^j_t = \log\left(\frac{S^j_t}{K}\right)$. The Fourier transform of the scaled spread call option is defined for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta_P = \{\boldsymbol{R} \in \mathbb{R}^d \mid R_j < -1, j = 1, \ldots, d, \text{ and } R_1 < -1 - \sum_{j=2}^{d} R_j\}$, by Eberlein et al. [2010]

$$\widehat{P}(\boldsymbol{z}) = \frac{\Gamma\left(\mathrm{i}\left(z_1 + \sum_{j=2}^{d} z_j\right) - 1\right)\prod_{j=2}^{d}\Gamma(-\mathrm{i}z_j)}{\Gamma(\mathrm{i}z_1 + 1)}. \tag{4.4}$$

Example 4.3 (Call on min)**.** The payoff function of the call on min option on d stocks is given by:

$$P(\boldsymbol{X}_T) = \max\left(\min(e^{X_T^1}, \ldots, e^{X_T^d}) - K, 0\right). \tag{4.5}$$

The payoff function can be straightforwardly scaled to become independent of the strike by defining $X_t^j = \log\left(\frac{S_t^j}{K}\right)$. The Fourier transform of the scaled call on min option is defined for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta_P = \{\boldsymbol{R} \in \mathbb{R}^d \mid R_j < 0,\ j = 1, \ldots, d, \text{ and } \sum_{j=1}^d R_j < -1\}$, by Eberlein et al. [2010]

$$\widehat{P}(\boldsymbol{z}) = \frac{1}{\left(\mathrm{i}\left(\sum_{j=1}^d z_j\right) - 1\right)\prod_{j=1}^d (\mathrm{i}z_j)}. \tag{4.6}$$

Example 4.4 (Put on max)**.** The payoff function of the put on max option on d stocks is given by:

$$P(\boldsymbol{X}_T) = \max\left(K - \max(e^{X_T^1}, \ldots, e^{X_T^d}), 0\right). \tag{4.7}$$

The payoff function can be straightforwardly scaled to become independent of the strike by defining $X_t^j = \log\left(\frac{S_t^j}{K}\right)$. The Fourier transform of the scaled put on max option is defined for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta_P = \{\boldsymbol{R} \in \mathbb{R}^d \mid R_j > 0,\ j = 1, \ldots, d\}$, by Eberlein et al. [2010]

$$\widehat{P}(\boldsymbol{z}) = \frac{(-1)^d}{\left(1 - \mathrm{i}\left(\sum_{j=1}^d z_j\right)\right)\prod_{j=1}^d (z_j)}. \tag{4.8}$$

Example 4.5 (Cash-Or-Nothing (CON) Put)**.** The payoff function of the cash-or-nothing put option on d stocks is given by

$$P(\boldsymbol{X}_T) = \prod_{j=1}^d \mathbb{1}_{\{e^{X_T^j} < K\}}(X_T^j). \tag{4.9}$$

The payoff function can be straightforwardly scaled to become independent of the strike by defining $X_t^j = \log\left(\frac{S_t^j}{K}\right)$. The Fourier transform of the scaled cash-or-nothing put option is defined for $\boldsymbol{z} = \boldsymbol{y} + \mathrm{i}\boldsymbol{R} \in \mathbb{C}^d$ with $\boldsymbol{R} \in \delta_P = \{\boldsymbol{R} \in \mathbb{R}^d \mid R_j > 0,\ j = 1, \ldots, d\}$, by Eberlein et al. [2010]

$$\widehat{P}(\boldsymbol{z}) = \prod_{j=1}^d \left(-\frac{1}{\mathrm{i}z_j}\right). \tag{4.10}$$

Remark 4.6. If the payoff function is scaled by the strike price (see Section 4), the integrand $g(\cdot)$ in (1.2) is multiplied by K.

5 Domain Transformations

This Section describes the distributions used for the domain transformation briefly explained in Section 2. The details related to each of the considered distributions are located in the file

`domain_transformation.py`. Each distribution has the following modules: `pdf` ($\psi_{\boldsymbol{Y}}(\cdot)$), `icdf` ($\Psi^{-1}_{\sqrt{W}}$), `decompose` (method for computing $\boldsymbol{L}$).

One of the main challenges of the domain transformation from $\mathbb{R}^d$ to $[0,1]^d$ is that the ICDF Ψ^{-1} is not given in closed form for most multivariate distributions, and efficient approximations are not always available. In the case of a multivariate normal distribution, we have that if $\boldsymbol{L}\boldsymbol{L}^T = \tilde{\boldsymbol{\Sigma}}$, then $\boldsymbol{Y} \sim \mathcal{N}_d(\boldsymbol{0}, \tilde{\boldsymbol{\Sigma}}) \overset{\mathrm{d}}{=} \boldsymbol{L}\boldsymbol{Z}$ with $\boldsymbol{Z} \sim \mathcal{N}_d(\boldsymbol{0}, \boldsymbol{I}_d)$, where " $\overset{\mathrm{d}}{=}$ ", should be understood as an equality in the distribution sense, and $\boldsymbol{L}$ can be obtained using, for instance, the Cholesky decomposition or the eigenvalue decomposition. The advantage of the latter representation is that the components of the random vector are uncorrelated, and hence independent, and the ICDF can be computed component-wise as follows $\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u}) = \boldsymbol{L}\left(\Psi^{-1}_{Z}(u_1), \ldots, \Psi^{-1}_{Z}(u_d)\right)^\top$ for $\boldsymbol{u} \in [0,1]^d$ with $Z \in \mathcal{N}(0,1)$. However, for multivariate distributions other than the multivariate normal distribution, uncorrelating the components of $\boldsymbol{Y}$ by factoring the covariance matrix does not imply mutual independence between the components.

In order to eliminate the dependence between the componets of $\boldsymbol{Y}$ for other multivariate distributions, we choose to transform the domain with distributions that belong to the class of (centered) normal mean-variance mixtures i.e. $\boldsymbol{Y} \overset{\mathrm{d}}{=} \sqrt{W}\boldsymbol{L}\boldsymbol{Z}$, with $\boldsymbol{Z} \sim \mathcal{N}_d(\boldsymbol{0}, \boldsymbol{I}_d)$ independent of W, a scalar random variable, and $\boldsymbol{L} \in \mathbb{R}^{d\times d}$. Then, by applying an affine transformation to the integral in (2.5), we obtain

$$\int_{[0,1]^d} \frac{g\left(\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u})\right)}{\psi_{\boldsymbol{Y}}\left(\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u})\right)} \mathrm{d}\boldsymbol{u} = \int_{[0,1]^{d+1}} \frac{g\left(\Psi^{-1}_{\sqrt{W}}(u_{d+1})\, \boldsymbol{L}\Psi^{-1}_{\boldsymbol{Z}}\left(\boldsymbol{u}_{-(d+1)}\right)\right)}{\psi_{\boldsymbol{Y}}\left(\Psi^{-1}_{\sqrt{W}}(u_{d+1})\, \boldsymbol{L}\Psi^{-1}_{\boldsymbol{Z}}\left(\boldsymbol{u}_{-(d+1)}\right)\right)} \mathrm{d}\boldsymbol{u},$$

where $\boldsymbol{u}_{-(d+1)} = (u_1, \ldots, u_d)$ represents the vector $\boldsymbol{u} \in [0,1]^{d+1}$ without the $(d+1)$-th component. On the other hand, if we have that $\boldsymbol{Y} \overset{\mathrm{d}}{=} \frac{1}{\sqrt{W}}\boldsymbol{L}\boldsymbol{Z}$, we obtain

$$\int_{[0,1]^d} \frac{g\left(\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u})\right)}{\psi_{\boldsymbol{Y}}\left(\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u})\right)} d\boldsymbol{u} = \int_{[0,1]^{d+1}} \frac{g\left(\left(\Psi^{-1}_{\sqrt{W}}(u_{d+1})\right)^{-1} \boldsymbol{L}\Psi^{-1}_{\boldsymbol{Z}}\left(\boldsymbol{u}_{-(d+1)}\right)\right)}{\psi_{\boldsymbol{Y}}\left(\left(\Psi^{-1}_{\sqrt{W}}(u_{d+1})\right)^{-1} \boldsymbol{L}\Psi^{-1}_{\boldsymbol{Z}}\left(\boldsymbol{u}_{-(d+1)}\right)\right)} d\boldsymbol{u}$$

For more details, we refer to Bayer et al. [2024].

Example 5.1 (Multivariate Normal (MN) Distribution)**.** The probability density function of the multivariate normal distribution with zero mean and covariance matrix $\tilde{\boldsymbol{\Sigma}}$ is given by:

$$\psi(\boldsymbol{y}) = (2\pi)^{-\frac{d}{2}} \left(\det(\tilde{\boldsymbol{\Sigma}})\right)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\boldsymbol{y}^\top \tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{y}\right)\right), \boldsymbol{y} \in \mathbb{R}^d.$$

When $\boldsymbol{Y}$ follows the MN distribution, the ICDF can be computed as explained above i.e. $\Psi^{-1}_{\boldsymbol{Y}}(\boldsymbol{u}) = \boldsymbol{L}\left(\Psi^{-1}_{Z}(u_1), \ldots, \Psi^{-1}_{Z}(u_d)\right)^\top$. In order to maintain the same structure of the implementation for all the transformation distributions, this can be viewed as having $W = 1$, a constant scalar random variable, and $\Psi^{-1}_{\sqrt{W}}(u_{d+1}) = 1$ for all $u_{d+1} \in [0,1]$.

Example 5.2 (Multivariate Student-t (MS) Distribution)**.** The probability density function of the multivariate Student-t distribution with zero mean, covariance matrix $\tilde{\boldsymbol{\Sigma}}$, and degrees of freedom $\tilde{\nu} > 0$ is given by:

$$\psi(\boldsymbol{y}) = \frac{\Gamma\left(\frac{\tilde{\nu}+d}{2}\right)\left(\det(\tilde{\boldsymbol{\Sigma}})\right)^{-\frac{1}{2}}}{\Gamma\left(\frac{\tilde{\nu}}{2}\right)\tilde{\nu}^{\frac{d}{2}}\pi^{\frac{d}{2}}}\left(1+\frac{1}{\tilde{\nu}}\left(\boldsymbol{y}^\top\tilde{\boldsymbol{\Sigma}}\boldsymbol{y}\right)\right)^{-\frac{\tilde{\nu}+d}{2}}, \boldsymbol{y}\in\mathbb{R}^d.$$

When $\boldsymbol{Y}$ follows the MS distribution i.e. $\mathbf{Y}\sim t_d(\mathbf{0},\tilde{\boldsymbol{\Sigma}},\tilde{\nu})$, then we have that $\boldsymbol{Y}\overset{d}{=}\sqrt{\frac{\tilde{\nu}}{W}}\boldsymbol{L}\boldsymbol{Z}$ with $\boldsymbol{Z}\sim\mathcal{N}_d(\mathbf{0},\boldsymbol{I}_d)$ and $W\sim\chi^2(\tilde{\nu})$, the chi-squared distribution with degrees of freedom equal to $\tilde{\nu}$, independent of $\boldsymbol{Z}$. $\sqrt{W}$ follows the chi distribution with degrees of freedom parameter equal to $\tilde{\nu}$.

Example 5.3 (Multivariate Laplace (ML) Distribution)**.** The probability density function of the multivariate Laplace distribution with zero mean and covariance matrix $\tilde{\boldsymbol{\Sigma}}$ is given by:

$$\psi(\boldsymbol{y}) = (2\pi)^{-\frac{d}{2}}(\det(\tilde{\boldsymbol{\Sigma}}))^{-\frac{1}{2}}\left(\frac{\boldsymbol{y}^\top\tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{y}}{2}\right)^{\frac{v}{2}}K_v\left(\sqrt{2\boldsymbol{y}^\top\tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{y}}\right), \boldsymbol{y}\in\mathbb{R}^d$$

where $v=\frac{2-d}{2}$. When $\boldsymbol{Y}$ follows the ML distribution i.e. $\mathbf{Y}\sim\mathrm{ML}_d(\mathbf{0},\tilde{\boldsymbol{\Sigma}})$, then we have that $\boldsymbol{Y}\overset{d}{=}\sqrt{W}\boldsymbol{L}\boldsymbol{Z}$ with $\boldsymbol{Z}\sim\mathcal{N}_d(\mathbf{0},\boldsymbol{I}_d)$ and $W\sim\exp(1)$, the exponential distribution with rate parameter equal to 1, independent of $\boldsymbol{Z}$. $\sqrt{W}$ follows the Rayleigh distribution with scale parameter equal to $\frac{1}{\sqrt{2}}$.

The implementation allows the user to choose between MN, MS and ML distributions for the transformation, along with their parameters. If the choice of the distribution is not specified, then the default choice is based on the boundary growth conditions provided in Bayer et al. [2024], which can be located in the file `pricing_engine.py`. The default choice for transformations is summarized in Table 5.1.

Model	**Transformation**	**Parameters**
GBM	MN	$\tilde{\boldsymbol{\Sigma}}=\frac{1}{T}\boldsymbol{\Sigma}^{-1}$
VG	MS	$\tilde{\boldsymbol{\Sigma}}=\boldsymbol{\Sigma}^{-1},\tilde{\nu}=\frac{2T}{\nu}-d$
NIG	ML	$\tilde{\boldsymbol{\Sigma}}=\frac{2}{\delta^2T^2}\boldsymbol{\Delta}^{-1}$
GH	ML	$\tilde{\boldsymbol{\Sigma}}=\frac{2}{\delta^2T^2}\boldsymbol{\Delta}^{-1}$

Table 5.1: Choice of $\psi(\cdot)$ and its parameters for different stock price models.

6 Usage Guide

The `main()` function orchestrates the setup and execution of the pricing engine.

Model and Payoff Selection

The model and payoff are specified as follows:

```
model_name = 'GH'
payoff_name = 'basket_put'
```

- `model_name`: Specifies the model used, here 'GH' (Generalized Hyperbolic). - `payoff_name`: Specifies the type of option payoff, here 'basket put'.

Parameter Definition

Defines the parameters of the model and the payoff

```
d = 2
S0 = 100 * np.ones(d)
K = 100
r = 0
T = 1.0
q = 0
alpha = 10
beta = -3 * np.ones(d)
delta = 0.2
DELTA = np.identity(d)
lambda_var = 1
weights = 1/d * np.ones(d)
option_params = [d, S0, K, r, T, q, alpha, beta, delta, DELTA,
   lambda_var]
```

Weights are defined for basket options.

Pricing Execution

The pricing engine is executed with specified parameters:

```
N_samples = 2**7
S_shifts = 30
transform_distribution = None
transform_params = None
price_estimate, error_estimate = QMC_fourier_pricing_engine(
model_name,
payoff_name,
option_params,
N_samples,
S_shifts,
transform_distribution=transform_distribution,
transform_params=transform_params,
weights=weights
)
```

- Specification of the transform distribution and its parameters is optional. The function prints the estimated price and statistical error.

Remark 6.1. Ensure that `option_params` are provided in the correct order for each model:

- **GH Model:** $d, \boldsymbol{S_0}, K, T, r, q, \alpha, \boldsymbol{\beta}, \delta, \boldsymbol{\Delta}, \lambda$
- **NIG Model:** $d, \boldsymbol{S_0}, K, T, r, q, \alpha, \boldsymbol{\beta}, \delta, \boldsymbol{\Delta}$
- **VG Model:** $d, \boldsymbol{S_0}, K, T, r, q, \boldsymbol{\Sigma}, \theta, \nu$
- **GBM Model:** $d, \boldsymbol{S_0}, K, T, r, q, \boldsymbol{\Sigma}$

Execution

The script executes the main function when run as a standalone program:

```
if __name__ == "__main__":
main()
```

This ensures that the code runs only when executed directly.

Variable	Type	Description
d	int	Number of assets
S0	numpy.ndarray	Initial stock prices
K	float	Strike price
r	float	Risk-free interest rate
T	float	Time to maturity
q	float	Dividend yield
alpha	float	Model-specific parameter
beta	numpy.ndarray	Model-specific parameter
delta	float	Model-specific parameter
DELTA	numpy.ndarray	Model-specific parameter
lambda_var	float	Model-specific parameter
weights	numpy.ndarray	Weights for basket option
N_samples	int	Number of QMC samples
S_shifts	int	Number of QMC digital shifts

Table 6.1: Parameter Descriptions

7 Numerical Examples

Model	**Payoff**	**Parameters**	**Reference Price**	**Statistical Error**
GBM	Basket Put	$d = 4, S_0^j = 100,\ K = 100,\ r = 0.1,$ $q = 0.05,\ T = 1,\ \sigma_j = 0.2,\ \boldsymbol{\Sigma} =$ $\mathrm{diag}(\boldsymbol{\sigma}), w_j = \frac{1}{d}, j = 1, \ldots, d$	1.8546	6×10^{-3}
VG	Call on Min	$d = 8, S_0^j = 100,\ K = 100,\ r = 0.1,$ $q = 0.05,\ T = 1,\ \sigma_j = 0.4,\ \boldsymbol{\Sigma} =$ $\mathrm{diag}(\boldsymbol{\sigma}),\ \nu = 0.001,\ \boldsymbol{\theta}_j = -0.3, j =$ $1, \ldots, d$	0.01236	1.65×10^{-5}
NIG	Spread Call	$d = 2, S_0^1 = 100, S_0^2 = 50,\ K = 50,$ $r = 0.1,\ q = 0.05,\ T = 1,\ \alpha = 15,$ $\boldsymbol{\Delta} = \boldsymbol{I}_d,\ \delta = 0.1,\ \boldsymbol{\beta} = (-3, -3).$	4.5872	1.62×10^{-2}
GH	Put on Max	$d = 6, S_0^j = 100,\ K = 100,\ r = 0.1,$ $q = 0.05,\ T = 1,\ \alpha = 10,\ \boldsymbol{\Delta} =$ $\boldsymbol{I}_d,\ \delta = 0.1,\ \beta_j = -3, \lambda = -1, j =$ $1, \ldots, d$	0.12506	3.56×10^{-2}

Table 7.1: Reference prices and their corresponding statistical error of RQMC with $N = 2^{10}$ QMC points and $S = 30$ randomizations, for different pricing models and payoff functions.

References

C. Bayer, C. Ben Hammouda, A. Papapantoleon, M. Samet, and R. Tempone. Optimal damping with a hierarchical adaptive quadrature for efficient Fourier pricing of multi-asset options in Lévy models. *Journal of Computational Finance*, 27(3), 2023.

C. Bayer, C. B. Hammouda, A. Papapantoleon, M. Samet, and R. Tempone. Quasi-Monte Carlo for efficient Fourier pricing of multi-asset options. *arXiv preprint arXiv:2403.02832*, 2024.

S.-C. T. Choi, F. J. Hickernell, R. Jagadeeswaran, M. J. McCourt, and A. G. Sorokin. Quasi-Monte Carlo software. In *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 23–47. Springer, 2020.

R. Cranley and T. N. Patterson. Randomization of number theoretic methods for multiple integration. *SIAM Journal on Numerical Analysis*, 13(6):904–914, 1976.

J. Dick, F. Y. Kuo, and I. H. Sloan. High-dimensional integration: the quasi-Monte Carlo way. *Acta Numerica*, 22:133–288, 2013.

E. Eberlein, K. Glau, and A. Papapantoleon. Analysis of Fourier transform valuation formulas and applications. *Applied Mathematical Finance*, 17(3):211–240, 2010.

T. R. Hurd and Z. Zhou. A Fourier transform method for spread option pricing. *SIAM Journal on Financial Mathematics*, 1(1):142–157, 2010.

C. Lanczos. A precision approximation of the gamma function. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 1(1):86–96, 1964.

E. Luciano and W. Schoutens. A multivariate jump-driven financial asset model. *Quantitative finance*, 6(5):385–402, 2006.

P. L'Ecuyer. *Randomized quasi-Monte Carlo: An introduction for practitioners*. Springer, 2018.

C. W. Oosterlee and L. A. Grzelak. *Mathematical modeling and computation in finance: with exercises and Python and MATLAB computer codes*. World Scientific, 2019.

K. Prause et al. *The generalized hyperbolic model: Estimation, financial derivatives, and risk measures*. PhD thesis, Citeseer, 1999.

I. M. Sobol', D. Asotsky, A. Kreinin, and S. Kucherenko. Construction and comparison of high-dimensional Sobol'generators. *Wilmott*, 2011(56):64–79, 2011.