Electronic Journal of Statistics
Vol. 16 (2022) 2432–2474
ISSN: 1935-7524
https://doi.org/10.1214/22-EJS2006

Noise inference for ergodic Lévy driven SDE*

Hiroki Masuda

Department of Mathematical Sciences, Faculty of Mathematics, Kyushu University
e-mail: hiroki@math.kyushu-u.ac.jp

Lorenzo Mercuri

Department of Economics, Management and Quantitative Methods, University of Milan
e-mail: lorenzo.mercuri@unimi.it

and

Yuma Uehara

Department of Mathematics, Faculty of Engineering Science, Kansai University
e-mail: y-uehara@kansai-u.ac.jp

Abstract: We study inference for the driving Lévy noise of an ergodic stochastic differential equation (SDE) model, when the process is observed at high-frequency and long time and when the drift and scale coefficients contain finite-dimensional unknown parameters. By making use of the Gaussian quasi-likelihood function for the coefficients, we derive a stochastic expansion for functionals of the unit-time residuals, which clarifies some quantitative effect of plugging in the estimators of the coefficients, thereby enabling us to take several inference procedures for the driving-noise characteristics into account. We also present new classes and methods available in YUIMA for the simulation and the estimation of a Lévy SDE model. We highlight the flexibility of these new advances in YUIMA using simulated and real data.

MSC2020 subject classifications: Primary 62F12, 62M20; secondary 60G51, 62-04.
Keywords and phrases: Noise Inference, SDE driven by a Lévy process, Gaussian quasi-likelihood function.

Received November 2021.

Contents

1 Introduction 2433
2 Preliminaries 2435
  2.1 Notations and conventions 2435
  2.2 Basic assumptions 2435
  2.3 Examples 2437
  2.4 Stepwise Gaussian quasi-likelihood estimation 2438

*This work was partly supported by JST CREST Grant Number JPMJCR14D7, Japan.

3 Theoretical results . . . 2441
3.1 Stochastic expansion of residual functional . . . 2441
3.2 M-estimation of noise parameter . . . 2445
3.3 Further remarks . . . 2452
3.3.1 Dimension of the processes . . . 2452
3.3.2 Model selection for $\mathcal{L}(J_1)$ . . . 2452
3.3.3 Setting of noise inference . . . 2452
4 Implementation . . . 2452
4.1 yuima.law: A new class for a mathematical description of the Lévy process . . . 2453
4.2 yuima.qmleLevy.incr: Estimation of an SDE driven by a Lévy pure jump process in yuima . . . 2455
5 Numerical examples . . . 2457
5.1 Univariate Lévy SDE model . . . 2457
5.2 Multivariate Lévy SDE model . . . 2461
5.3 Real data . . . 2465
Acknowledgments . . . 2473
References . . . 2473

1. Introduction

We consider the following univariate Markovian stochastic differential equation (SDE):

$$dX_t = a(X_t, \alpha)dt + c(X_{t-}, \gamma)dJ_t^\eta, \tag{1}$$

where:

- The coefficients a and c are smooth enough with c being non-degenerate, and known except for an unknown parameter

$$\theta := (\gamma, \alpha) \in \Theta_\gamma \times \Theta_\alpha =: \Theta \subset \mathbb{R}^p$$

  for bounded convex domains $\Theta_\alpha \subset \mathbb{R}^{p_\alpha}$ and $\Theta_\gamma \subset \mathbb{R}^{p_\gamma}$ $(p = p_\alpha + p_\gamma)$;
- The driving noise J^η is a standardized non-Gaussian Lévy process with finite moments, whose distribution depends on an unknown parameter $\eta \in \Theta_\eta$, a domain in $\mathbb{R}^{p_\eta}$, and where J^η is independent of X_0.

We write $J = J^\eta$ in the sequel. We suppose that there are true values $\theta_0 = (\gamma_0, \alpha_0) \in \Theta_\gamma \times \Theta_\alpha$ and $\eta_0 \in \Theta_\eta$ which induce the true image measure P of (X, J), and that we observe a discrete-time sample $\boldsymbol{X}_n := (X_{t_j})_{j=1}^n$, where $t_j = t_j^n := jh_n$, with the sampling stepsize $h = h_n \to 0$ satisfying that

$$T_n := nh \to \infty \ \text{ and } nh^2 \to 0 \text{ as } n \to \infty, \tag{2}$$

the so-called rapidly increasing experimental design.

Our objective is to estimate the value $\xi_0 := (\theta_0, \eta_0)$ under the ergodicity. We remark the parameter may not completely characterize the distribution $\mathcal{L}(J)$, so

that the problem is not necessarily parametric; for example, η is just a skewness or kurtosis, which may or may not completely determine $\mathcal{L}(J_1)$.

For the estimation of the coefficient parameter θ, from the statistical point of view it is important what kind of distributional is used to approximate the conditional distribution of X_{t_j} given $X_{t_{j-1}}$. Previously, [12] and [18] considered estimation of θ based on the Gaussian quasi-likelihood (GQL), and proved the asymptotic normality and the tail probability estimates of the Gaussian quasi-maximum likelihood estimator (GQMLE). At the expense of efficiency, the GQL based method has the robustness against misspecification of the driving-noise distribution, which may be crucial in the context of time-series models, see [17, Section 6.2]. Further, concerned with estimation of a Lévy-measure functional of the form $\int \varphi(z)\nu(dz)$, with ν denoting the Lévy measure of J, the previous study [13] proposed a moment-matching based estimator and proved its asymptotic normality at rate $\sqrt{T_n}$. However, the procedure imposed some stringent conditions on the behavior of φ around the origin [13, Assumption 2.7], and was not quite suitable if we want to estimate $\mathcal{L}(J_1)$ directly; in general, estimation of the distribution and that of the corresponding Lévy measure can be of technically rather different matters.

In this paper, we will propose yet another strategy for estimating η based on the unit-time approximation, which goes as follows:

1. First we construct the GQMLE $\hat{\theta}_n := (\hat{\gamma}_n, \hat{\alpha}_n)$ and the residual
$$\widehat{\Delta_j J} = \widehat{\Delta_j J}^n := \frac{X_{t_j} - X_{t_{j-1}} - ha(X_{t_{j-1}}, \hat{\alpha}_n)}{c(X_{t_{j-1}}, \hat{\gamma}_n)}. \tag{3}$$
2. We then estimate the $\mathcal{L}(J_1)$-i.i.d. sequence $(i = 1, \ldots, n)$
$$\epsilon_i := J_i - J_{i-1}$$
by adding up the finer increments over the ith unit-time interval $[i-1, i]$:
$$\hat{\epsilon}_i := \sum_{j \in A_i} \widehat{\Delta_j J}, \tag{4}$$
where, for each $i \in \{1, 2, \ldots, \lfloor T_n \rfloor\}$,
$$A_i := \{j \in \mathbb{N} : i - 1 < t_j \le i\} = \left\{ j \in \mathbb{N} : \left\lfloor \frac{i-1}{h} \right\rfloor + 1 \le j \le \left\lfloor \frac{i}{h} \right\rfloor \right\},$$
and then measure the discrepancy between a functional of $\{\hat{\epsilon}_i\}$ and that of $\{\epsilon_i\}$ through a stochastic expansion.
3. For an appropriate function m, we estimate η by
$$\hat{\eta}_n \in \operatorname*{argmax}_{\eta \in \overline{\Theta_\eta}} \sum_{i=1}^{\lfloor T_n \rfloor} m\left(\hat{\epsilon}_i, \eta\right).$$

The rest of this paper is organized as follows. We briefly summarize some prerequisites in Section 2, and then presents theoretical results in Section 3. Section 4 introduces new classes and methods in YUIMA R package ([2] and [7]) for the estimation procedure proposed in the previous sections. Some numerical examples based on simulated and real data are given in Section 5.

2. Preliminaries

2.1. Notations and conventions

Here are some basic notations and conventions used in this paper.

- For any vector variable $x = (x^{(i)})$, we write $\partial_x = \left(\frac{\partial}{\partial x^{(i)}}\right)_i$.
- C denotes a universal positive constant which may vary at each appearance.
- $\top$ stands for the transpose operator, and $v^{\otimes 2} := vv^\top$ for any matrix v.
- For a matrix $M = (M_{ij}) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$ and vector $u = (u_j) \in \mathbb{R}^{d_2}$, we will write $M[u] = \sum_{j=1}^{d_2} M_{\cdot j} u_j \in \mathbb{R}^{d_1}$. In particular, when $d_1 = 1$, it stands for the dot product of two vectors. We will also write $M[U] = \sum_{k,l} M_{kl} U_{kl}$ for two square matrices M and U of the same order.
- The convergences in probability and in distribution are denoted by $\xrightarrow{p}$ and $\xrightarrow{\mathcal{L}}$, respectively, and all limits appearing below are taken for $n \to \infty$ unless otherwise mentioned.
- For two nonnegative real sequences (a_n) and (b_n), we write $a_n \lesssim b_n$ if $\limsup_n (a_n/b_n) < \infty$.
- For any $x \in \mathbb{R}$, $\lfloor x \rfloor$ denotes the maximum integer which does not exceed x.
- Given a function $h : \mathbb{R} \to \mathbb{R}^+$ and a signed measure m on a one-dimensional Borel space, we write

$$||m||_h = \sup \{|m(f)| : f \text{ is } \mathbb{R}\text{-valued, } m\text{-measurable and satisfies } |f| \le h\}.$$

2.2. Basic assumptions

Denote by $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\ge 0}, P)$ the underlying complete filtered probability space; every processes are adapted to the filtration $(\mathcal{F}_t)$. We will write E for the expectation operator associated with P.

Assumption 2.1. *The Lévy process J has moments of any order with $E[J_1] = 0$, $E[J_1^2] = 1$, and $E[|J_1|^q] < \infty$, for any $q > 0$.*

Assumption 2.2. *1. The drift coefficient $a(\cdot, \alpha_0)$ and scale coefficient $c(\cdot, \gamma_0)$ are Lipschitz continuous, and $c(x, \gamma) \neq 0$ for every (x, γ).*

2. For each $i \in \{0, 1\}$ and $k \in \{0, \dots, 4\}$, the following conditions hold:

- *The coefficients a and c admit the partial derivatives $\partial_x^i \partial_\alpha^k a$ and $\partial_x^i \partial_\gamma^k c$ for $i \ge 0$ and $k \ge 0$, all of which have continuous extensions as elements in $\mathcal{C}(\mathbb{R} \times \overline{\Theta})$.*

- *There exists nonnegative constant $C_{i,k}$ satisfying*

$$\sup_{(x,\alpha,\gamma)\in\mathbb{R}\times\overline{\Theta}_\alpha\times\overline{\Theta}_\gamma} \frac{1}{1+|x|^{C_{i,k}}}\left\{|\partial_x^i\partial_\alpha^k a(x,\alpha)|+|\partial_x^i\partial_\gamma^k c(x,\gamma)|+|c^{-1}(x,\gamma)|\right\}<\infty. \tag{5}$$

Then

Assumption 2.3. *1. There exists a probability measure π_0 such that for every $q>0$, we can find constants $a>0$ and $C_q>0$ for which*

$$\sup_{t\in\mathbb{R}_+}\exp(at)||P_t(x,\cdot)-\pi_0(\cdot)||_{h_q}\le C_q h_q(x), \tag{6}$$

for any $x\in\mathbb{R}$ where $h_q(x):=1+|x|^q$.

2. For any $q>0$, we have

$$\sup_{t\in\mathbb{R}_+}E[|X_t|^q]<\infty. \tag{7}$$

See [12, Proposition 5.4] for easy-to-verify conditions for Assumption 2.3.

We introduce a block-diagonal $p\times p$-matrix

$$\Gamma=\mathrm{diag}(\Gamma_\gamma,\Gamma_\alpha),$$

whose components are defined by:

$$\Gamma_\gamma:=2\int_{\mathbb{R}}\frac{(\partial_\gamma c(x,\gamma_0))^{\otimes 2}}{c^2(x,\gamma_0)}\pi_0(dx),$$
$$\Gamma_\alpha:=\int_{\mathbb{R}}\frac{(\partial_\alpha a(x,\alpha_0))^{\otimes 2}}{c^2(x,\gamma_0)}\pi_0(dx).$$

Assumption 2.4. Γ *is positive definite.*

We define real valued functions $\mathbb{H}_1(\gamma)$ and $\mathbb{H}_2(\alpha)$ on Θ_γ and Θ_α by

$$\mathbb{H}_1(\gamma)=-\int_{\mathbb{R}}\left(\log c^2(x,\gamma)+\frac{c^2(x,\gamma_0)}{c^2(x,\gamma)}\right)\pi_0(dx), \tag{8}$$
$$\mathbb{H}_2(\alpha)=-\int_{\mathbb{R}}c(x,\gamma_0)^{-2}(a(x,\alpha_0)-a(x,\alpha))^2\pi_0(dx). \tag{9}$$

We assume the following identifiability condition for $\mathbb{H}_1(\gamma)$ and $\mathbb{H}_2(\alpha)$:

Assumption 2.5. $\theta^\star\in\Theta$*, and there exist positive constants χ_γ and χ_α such that for all $(\gamma,\alpha)\in\Theta$,*

$$\mathbb{Y}_1(\gamma):=\mathbb{H}_1(\gamma)-\mathbb{H}_1(\gamma_0)\le-\chi_\gamma|\gamma-\gamma_0|^2, \tag{10}$$
$$\mathbb{Y}_2(\alpha):=\mathbb{H}_2(\alpha)-\mathbb{H}_2(\alpha_0)\le-\chi_\alpha|\alpha-\alpha_0|^2. \tag{11}$$

2.3. Examples

Although Assumption 2.1 imposes a strong restriction on the mean and variance structure of Z, there is still room for statistical modeling of Z with respect to, for example, its skewness and jump activity. In this section, with the parameter constraints for Assumption 2.1, we give concrete Lévy processes induced from subordinators (i.e. non-decreasing Lévy processes) by the following two procedures: For given two independent subordinators τ^1 and τ^2, one can easily construct a possibly skewed Lévy process of finite variation by taking its bilateral version: $\tau' := \tau^1 - \tau^2$. Another way to deliver a new Lévy process is to take a normal mean variance mixture of a subordinator: for $\mu, \beta \in \mathbb{R}$, a subordinator τ, and a standard normal random variable η being independent of τ, the normal mean variance mixture of τ at time t is given by

$$Z_t = \mu t + \tau_t \beta + \sqrt{\tau_t}\eta.$$

By their construction, $E[|\tau'|^q] < \infty$ and $E[|Z|^q] < \infty$ hold as long as the q-th moment of τ^1, τ^2 and τ exists. It is worth noting that having a generator of τ^1, τ^2 and τ in hand, we can directly obtain that of τ' and Z. All of the following induced Lévy processes can be generated by the functions `setLaw` and `simulate` in YUIMA package.

Example 2.6. *(Bilateral gamma [9]) For $\delta_1, \gamma_1, \delta_2, \gamma_2 \in \mathbb{R}_+$, the bilateral gamma process τ' is defined by the difference of two independent gamma subordinators τ^1 and τ^2 whose Lévy densities are expressed as: for each $i \in \{1, 2\}$,*

$$f_{\tau^i}(z) = \frac{\delta_i}{z} e^{-\gamma_i z}, \quad z > 0.$$

We write the law of τ'_1 as $bgamma(\delta_1, \gamma_1, \delta_2, \gamma_2)$ and it is straightforward from the form of f_{τ^i} that $\tau'_t \sim bgamma(\delta_1 t, \gamma_1, \delta_2 t, \gamma_2)$. Since the density function of $bgamma(\delta_1, \gamma_1, \delta_2, \gamma_2)$ is the convolution of two gamma density, it satisfies the symmetry relation

$$p(x; \delta_1, \gamma_1, \delta_2, \gamma_2) = p(-x; \delta_2, \gamma_2, \delta_1, \gamma_1), \quad x \in \mathbb{R} \setminus \{0\},$$

and on the positive real line, its form is given by

$$\begin{aligned} p(x; \delta_1, \gamma_1, \delta_2, \gamma_2) &= \frac{\gamma_1^{\delta_1}\gamma_2^{\delta_2}}{(\gamma_1+\gamma_2)^{\frac{1}{2}(\delta_1+\delta_2)}\Gamma(\delta_1)} x^{\frac{1}{2}(\delta_1+\delta_2)-1} e^{-\frac{x}{2}(\gamma_1-\gamma_2)} \\ &\times W_{\frac{1}{2}(\delta_1-\delta_2), \frac{1}{2}(\delta_1+\delta_2-1)}(x(\gamma_1+\gamma_2)), \end{aligned}$$

where $W_{\lambda,\mu}(z)$ denotes the Whittaker function. By using the independence between τ^1 and τ^2, the parameter constraints for Assumption 2.1 are written as follows:

$$E[Z_1] = \frac{\delta_1}{\gamma_1} - \frac{\delta_2}{\gamma_2} = 0, \quad V[Z_1] = \frac{\delta_1}{\gamma_1^2} + \frac{\delta_2}{\gamma_2^2} = 1.$$

Example 2.7. *(Normal (exponentially) tempered stable) The normal (exponentially) tempered stable law $NTS(\alpha, a, b, \beta, \mu)$ is defined by the law of the normal mean variance mixture of the positive exponentially tempered stable random variable whose Lévy density is given by*

$$f(z) = az^{-1-\alpha}e^{-bz}, \quad \alpha \in (0,1),\ a > 0,\ b > 0,$$

and its law is denoted by $TS(\alpha, a, b)$; especially inverse Gaussian law corresponds to $\alpha = \frac{1}{2}$. From [15, Theorem 30.1], the Lévy density of $NTS(\alpha, a, b, \beta, \mu)$ is explicitly expressed as:

$$g(z) = \sqrt{\frac{2}{\pi}} a e^{\beta z} \left(\frac{z^2}{2b + \beta^2} \right)^{-\frac{\alpha}{2} - \frac{1}{4}} K_{\alpha + \frac{1}{2}} \left(z \sqrt{(2b + \beta^2)} \right),$$

where $K_{\alpha+\frac{1}{2}}$ stands for the modified Bessel function of the third kind with index $\alpha + \frac{1}{2}$. From the expression of the Lévy density, the associated positive exponentially tempered stable subordinator τ satisfies $\tau_t \sim TS(\alpha, at, b)$ and thus the corresponding normal (exponentially) tempered stable process Z also does $Z_t \sim NTS(\alpha, at, b, \beta, \mu t)$. Since $z^\lambda K_\lambda(z) \lesssim 1$ as $z \downarrow 0$ for $\lambda > 0$, the Blumenthal-Getoor index of Z is 2α. The parameter constraints for Assumption 2.1 are written as follows:

$$E[Z_1] = \mu - a\alpha\Gamma(-\alpha)b^{\alpha-1}\beta = 0, \quad V[Z_1] = a\alpha\Gamma(-\alpha)b^{\alpha-1}\left[\frac{(\alpha-1)\beta^2}{b} - 1\right] = 1.$$

Especially in the simple case where $\mu = \beta = 0$ (that is, Z is a time-changed Brownian motion), the above constraints are reduced to

$$-a\alpha b^{\alpha-1}\Gamma(-\alpha) = 1.$$

2.4. Stepwise Gaussian quasi-likelihood estimation

Here and in what follows, for any process Y we will denote by $\Delta_j Y$ the j-th increment

$$\Delta_j Y := Y_{t_j} - Y_{t_{j-1}},$$

and $f_{j-1}(\theta) := f(X_{t_{j-1}}, \theta)$ for a measurable function f on $\mathbb{R} \times \overline{\Theta}$. Building on the discrete time formal Gaussian approximation, we define the stepwise GQL functions $\mathbb{H}_{1,n}(\gamma)$ and $\mathbb{H}_{2,n}(\alpha)$ as follows [18]:

$$\mathbb{H}_{1,n}(\gamma) := -\frac{1}{2T_n} \sum_{j=1}^{n} \left(h \log c_{j-1}^2(\gamma) + \frac{(\Delta_j X)^2}{c_{j-1}^2(\gamma)} \right),$$

$$\mathbb{H}_{2,n}(\alpha) := -\frac{1}{2T_n} \sum_{j=1}^{n} \frac{(\Delta_j X - h a_{j-1}(\alpha))^2}{h c_{j-1}^2(\hat{\gamma}_n)},$$

where $\hat{\gamma}_n$ is any maximizer of $\mathbb{H}_{1,n}$ over $\overline{\Theta}_\gamma$. We then define the associated stepwise GQMLE $\hat{\theta}_n := (\hat{\gamma}_n, \hat{\alpha}_n)$ where $\hat{\alpha}_n$ is any maximizer of $\mathbb{H}_{2,n}$ over $\overline{\Theta}_\alpha$.

Formally, the first-stage $\mathbb{H}_{1,n}(\gamma)$ corresponds to the quasi log-likelihoods associated with the (fake) approximation $N(x,\, c^2(x,\gamma)h)$ for $\mathcal{L}(X_{t_j}|X_{t_{j-1}} = x)$, and also the second-stage $\mathbb{H}_{2,n}(\alpha)$ does to the one associated with $N(x + a(x,\alpha)h,\, c^2(x,\hat{\gamma}_n)h)$.

Let $\nu_0(dz)$ denote the (true) Lévy measure of J, and

$$\nu_k := \int z^k\, \nu_0(dz), \qquad k \geq 2.$$

Under the aforementioned Assumptions 2.1 to 2.5, both ν_3 and ν_4 exist and are finite (and $E[J_1^2] = \nu_2 + \sigma^2 = 1$, where $\sigma^2 \geq 0$ denotes the Gaussian variance of J, possibly $\sigma^2 = 0$). We can deduce the asymptotic normality and the uniform tail-probability estimate:

Theorem 2.8. *Suppose that Assumptions 2.1 to 2.5 holds.*

1. *The GQMLE $\hat{\theta}_n = (\hat{\gamma}_n, \hat{\alpha}_n)$ satisfies*

$$\sqrt{T_n}(\hat{\theta}_n - \theta_0) \xrightarrow{\mathcal{L}} N_{p_\gamma + p_\alpha}\left(0, \Gamma^{-1}\Sigma_\theta(\Gamma^{-1})^\top\right), \tag{12}$$

where the $p \times p$-matrix

$$\Sigma_\theta = \begin{pmatrix} \Sigma_\gamma & \Sigma_{\gamma,\alpha} \\ \Sigma_{\gamma,\alpha}^\top & \Sigma_\alpha \end{pmatrix},$$

is defined by

$$\Sigma_\gamma = \nu_4 \int \left(\frac{\partial_\gamma c(x,\gamma_0)}{c(x,\gamma_0)}\right)^{\otimes 2} \pi_0(dx),$$
$$\Sigma_{\gamma,\alpha} = \nu_3 \int \frac{\partial_\gamma c(x,\gamma_0)\partial_\alpha a(x,\alpha_0)}{c^2(x,\gamma_0)} \pi_0(dx),$$
$$\Sigma_\alpha = \int \left(\frac{\partial_\alpha a(x,\alpha_0)}{c(x,\gamma_0)}\right)^{\otimes 2} \pi_0(dx).$$

2. *For any $L > 0$, there exists a constant $C_L > 0$ such that*

$$\sup_{n\in\mathbb{N}} P\left(\left|\sqrt{T_n}(\hat{\theta}_n - \theta_0)\right| > r\right) \leq \frac{C_L}{r^L}, \qquad r > 0.$$

We note that $\Sigma_{\gamma,\alpha}$, $\hat{\alpha}_n$ and $\hat{\gamma}_n$ are asymptotically independent if $\nu_3 = 0$, hence if in particular ν is symmetric. We refer to [12] and [18] for technical details of the proof of Theorem 2.8; although the two cited papers used a Z-estimator type identifiability condition, which is seemingly different from Assumption 2.5 (M-estimator type), it is trivial that we can follow the same line without any essential change.

To appreciate the difficulty of relaxing the standing condition $nh^2 \to 0$ in (2), let us first mention the case of diffusions: let (w_t) be a standard Wiener process, and consider the following one-dimensional diffusion process

$$dY_t = a(Y_t)dt + b(Y_t)dw_t$$

defined on the stochastic bases $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in\mathbb{R}_+}, P)$. Write the infinitesimal generator of Y as $\mathcal{A}$. By repeatedly applying Itô's formula, it follows that for $0 \le s < t$, $q \in \mathbb{N}$, and a sufficiently smooth function f, we have

$$
\begin{aligned}
&f(Y_t)\\
&= f(Y_s) + \int_s^t \partial f(Y_u) dY_u + \frac{1}{2}\int_s^t \partial^2 f(Y_u) b^2(Y_u) du\\
&= f(Y_s) + \int_s^t \left(\partial f(Y_u) a(Y_u) + \frac{1}{2}\partial^2 f(Y_u) b^2(Y_u)\right) du + \int_s^t \partial f(Y_u) b(Y_u) dw_u\\
&= f(Y_s) + \int_s^t \mathcal{A} f(Y_u) du + \int_s^t \partial f(Y_u) b(Y_u) dw_u\\
&= f(Y_s) + \int_s^t \left(\mathcal{A} f(Y_s) + \int_u^s \mathcal{A}^2 f(Y_v) dv + \int_u^s \partial \mathcal{A} f(Y_v) b(Y_v) dw_v\right) du\\
&+ \int_s^t \partial f(Y_u) b(Y_u) dw_u\\
&= f(Y_s) + (t-s)\mathcal{A} f(Y_s) + \int_s^t \left(\int_u^s \mathcal{A}^2 f(Y_v) dv + \int_u^s \partial \mathcal{A} f(Y_v) b(Y_v) dw_v\right) du\\
&+ \int_s^t \partial f(Y_u) b(Y_u) dw_u\\
&= \sum_{i=0}^{q-1} \frac{(t-s)^i}{i!} \mathcal{A}^i f(Y_s) + \int_s^t \int \dots \int \mathcal{A}^q f(Y_{v_q}) dv_1 \dots dv_q + (\text{martingale term}).
\end{aligned}
$$

Thus, under suitable integrability conditions, we obtain the expansion of $E[f(Y_t)|\mathcal{F}_s]$:

$$
E[f(Y_t)|\mathcal{F}_s] = \sum_{i=0}^{q-1} \frac{(t-s)^i}{i!} \mathcal{A}^i f(Y_s) + O_p((t-s)^q). \tag{13}
$$

In particular, the first-order approximation of the conditional expectation and conditional variance are given by

$$
\begin{aligned}
&E[X_t|\mathcal{F}_s] \approx X_s + (t-s) a(X_s, \alpha_0),\\
&V[X_t|\mathcal{F}_s] \approx E[(X_t - X_s - (t-s)a(X_s, \alpha_0))^2|\mathcal{F}_s] \approx (t-s) c(X_s, \gamma_0),
\end{aligned}
$$

respectively, which are used for constructing the GQL for diffusions. Relaxation of the condition $nh^2 \to 0$ to $nh^k \to 0$ for $k > 2$ is then possible by taking $q = q(k)$ large enough according to the value of k: the associated GQMLE has the consistency and asymptotic normality under $nh^k \to 0$. We refer to [8] for details. At this point, we should remark that the GQL based on the first term on the right-hand side of (13) (with $f(y) = y$ and $f(y) = y^2$) is fully explicit whatever $k > 2$ is.

On the other hand, although similar Itô-Taylor expansions to $E[X_t|\mathcal{F}_s]$ and $V[X_t|\mathcal{F}_s]$ can be easily derived in our Lévy driven case (1), the corresponding

infinitesimal generator contains not only the differential operator but also the integral operator with respect to the Lévy measure ν_η of the driving Lévy noise. Specifically, its infinitesimal generator $\tilde{\mathcal{A}}$ is given by

$$\tilde{\mathcal{A}}f(x) = a(x,\alpha)\partial f(x) + \int (f(x + c(x,\gamma)z) - f(x) - \partial f(x)c(x,\gamma)z)\nu_\eta(dz),$$

for a suitable function f. Consequently, the modified GQL based on the higher-order Itô-Taylor expansion (13) contains the unknown parameter η in addition to the drift and scale parameters. It is not clear that the simultaneous estimation of α, γ, and η by the modified GQL has a nice theoretical property. Even if it does, the entailed numerical optimization involved would be quite heavy and unstable since, for each η, we need to repeatedly compute several integrals with respect to ν_η inside of the modified GQL. For this reason, it is difficult to remove the condition $nh^2 \to 0$ in the present general non-linear-SDE setting, as long as we use the GQL based on the stochastic expansion (13).

In practice, the sampling points $t_1, \dots t_n$ may not be equally spaced. In such a case, supposing (2), we remark that the same statement as in Theorem 2.8 remains in place under the additional "sampling-balance" condition:

$$\frac{\min_{1\le j\le n}(t_j - t_{j-1})}{\max_{1\le j\le n}(t_j - t_{j-1})} \to 1.$$

For more technical details, see the discussion in [12, p. 1604–1605].

3. Theoretical results

3.1. Stochastic expansion of residual functional

Having the GQMLE in hand, we now turn to approximating $\mathcal{L}(J_1)$, the the unit-time distribution of J, based on the residuals

$$\hat{\epsilon}_i = \sum_{j\in A_i} \widehat{\Delta_j J},$$

where

$$\widehat{\Delta_j J} = \frac{\Delta_j X - h a_{j-1}(\hat{\alpha}_n)}{c_{j-1}(\hat{\gamma}_n)}.$$

Write $\hat{u}_{\alpha,n} = \sqrt{T_n}(\hat{\alpha}_n - \alpha_0)$ and $\hat{u}_{\gamma,n} = \sqrt{T_n}(\hat{\gamma}_n - \gamma_0)$, and let

$$\hat{u}_{\theta,n} := (\hat{u}_{\gamma,n}, \hat{u}_{\alpha,n}).$$

From now on we will mostly omit "(θ_0)" from notation. In particular, for a measurable function $f(x,\theta)$ we will abbreviate $f_{j-1}(\theta_0)$ as f_{j-1}.

Theorem 3.1. *Suppose that Assumptions 2.1 to 2.5 hold. Let $\rho : \mathbb{R} \to \mathbb{R}$ be a $\mathcal{C}^2$-function such that*

$$\max_{i\in\{0,1,2\}} \left|\partial_\epsilon^i \rho(\epsilon)\right| \lesssim 1 + |\epsilon|^C.$$

Then, we have

$$\sum_{i=1}^{\lfloor T_n \rfloor} \rho\left(\hat{\epsilon}_i\right) = \sum_{i=1}^{\lfloor T_n \rfloor} \rho\left(\epsilon_i\right) + \frac{1}{\sqrt{T_n}} \sum_{i=1}^{\lfloor T_n \rfloor} \partial_\epsilon \rho\left(\epsilon_i\right) b_i[\hat{u}_{\theta,n}] + o_p\left(\sqrt{T_n}\right), \tag{14}$$

where the random sequence $(b_i)_{i=1}^{\lfloor T_n \rfloor} \subset \mathbb{R}^p$ *is given by*

$$b_i = b_i(\theta_0) := \sum_{j \in A_i} \begin{pmatrix} \partial_\gamma (c^{-1})_{j-1} (\Delta_j X - h a_{j-1}) \\ -h c_{j-1}^{-1} \partial_\alpha a_{j-1} \end{pmatrix}.$$

Theorem 3.1 reveals the quantitative effect of plugging in $\hat{\theta}_n$. The second term on the right-hand side of (14) is not $o_p(\sqrt{T_n})$ but $O_p(\sqrt{T_n})$, which implies that the asymptotic distribution of $\hat{\eta}_n$ is indeed subject to influence of the proposed unit-time approximation; this is natural and expected, for we are using the $\sqrt{T_n}$-consistent (generally sub-optimal) estimator of θ.

To prove Theorem 3.1, we begin with a preliminary estimate.

Lemma 3.2. *Under Assumptions 2.1 to 2.5, for each* $r \geq 2$ *we have*

$$\max_{1 \leq i \leq \lfloor T_n \rfloor} E\left[\left|\hat{\epsilon}_i - \epsilon_i - \frac{1}{\sqrt{T_n}} b_i[\hat{u}_{\theta,n}]\right|^r\right] \lesssim h \vee T_n^{-r}. \tag{15}$$

Proof. We will abbreviate $a(X_s, \alpha_0)$ as a_s and so on; with a slight abuse of notation, we will write a_{j-1} for $a_{t_{j-1}}$. Write $\widehat{\Delta_j J} = \delta_j(\hat{\theta}_n)$; then,

$$\delta_j(\hat{\theta}_n) = \delta_j + (\partial_\theta \delta_j)[\hat{\theta}_n - \theta_0] + \frac{1}{2}\left(\partial_\theta^2 \delta_j(\theta_0 + s_n(\hat{\theta}_n - \theta_0))\right)[(\hat{\theta}_n - \theta_0)^{\otimes 2}]$$

for a suitable random point $s_n \in [0,1]$. Decompose $\hat{\epsilon}_i - \epsilon_i$ as follows:

$$\begin{aligned} \hat{\epsilon}_i - \epsilon_i &= \sum_{j \in A_i} (\widehat{\Delta_j J} - \Delta_j J) + \left(J_{(\lfloor \frac{i-1}{h} \rfloor + 1)h} - J_{i-1}\right) + \left(J_i - J_{\lfloor \frac{i}{h} \rfloor h}\right) \\ &= \frac{1}{\sqrt{T_n}} b_i[\hat{u}_{\theta,n}] + \zeta_{1,i} + \zeta_{2,i} + \zeta_{3,i}, \end{aligned}$$

where

$$\begin{aligned} \zeta_{1,i} &:= \frac{1}{2T_n} \sum_{j \in A_i} \left(\partial_\theta^2 \delta_j(\theta_0 + s_n(\hat{\theta}_n - \theta_0))\right)[\hat{u}_{\theta,n}^{\otimes 2}], \\ \zeta_{2,i} &:= \sum_{j \in A_i} (\delta_j - \Delta_j J) = \sum_{j \in A_i} c_{j-1}^{-1} \left(\int_{t_{j-1}}^{t_j} (a_s - a_{j-1}) ds \right. \\ &\qquad \left. + \int_{t_{j-1}}^{t_j} (c_{s-} - c_{j-1}) dJ_s\right), \\ \zeta_{3,i} &:= \left(J_{(\lfloor \frac{i-1}{h} \rfloor + 1)h} - J_{i-1}\right) + \left(J_i - J_{\lfloor \frac{i}{h} \rfloor h}\right). \end{aligned}$$

Before proceeding, let us note that by Theorem 2.8 the sequence $(\hat{u}_{\theta,n})$ is L^r-bounded for each $r \geq 2$:

$$E\left(|\hat{u}_{\theta,n}|^r\right) \lesssim 1, \tag{16}$$

where we implicitly assume that $r \geq 1$ when using this notation.

First we will deduce

$$\max_{1\leq i\leq \lfloor T_n \rfloor} E\left(|\zeta_{1,i}|^r\right) \lesssim T_n^{-r}. \tag{17}$$

Since the parameter space $\Theta = \Theta_\alpha \times \Theta_\gamma$ is supposed to be bounded and convex, the Sobolev inequality is in force (see [1] for details): for a random field $u \in \mathcal{C}^1(\Theta)$ and $q > p$, we have

$$E\left[\sup_{\theta\in\Theta} |u(\theta)|^q\right] \lesssim \sup_{\theta\in\Theta} \left\{E[|u(\theta)|^q] + E[|\partial_\theta u(\theta)|^q]\right\}.$$

Noting the identities

$$\begin{aligned}\partial_\gamma^k \partial_\alpha^l \delta_j(\theta) &= -h\left\{\partial_\gamma^k (c^{-1})_{j-1}(\gamma)\right\} \partial_\alpha^l a_{j-1}(\alpha),\\ \partial_\gamma^k \delta_j(\theta) &= \left\{\partial_\gamma^k (c^{-1})_{j-1}(\gamma)\right\} (\Delta_j X - h a_{j-1}(\alpha)),\end{aligned}$$

valid for each $k \geq 0$ and $l \geq 1$, we can apply Sobolev's and Jensen's inequalities to conclude that, for $r > p_\gamma$,

$$\begin{aligned}&E\left[\left|\sum_{j\in A_i}\left(\partial_\theta^2 \delta_j(\theta_0 + s_n(\hat{\theta}_n - \theta_0))\right)\right|^r\right]\\ &\lesssim E\left[\sup_{\gamma\in\Theta_\gamma}\left|\sum_{j\in A_i} \partial_\gamma^2 (c^{-1})_{j-1}(\gamma) \int_{t_{j-1}}^{t_j} c_{s-} dJ_s\right|^r\right] + E\left[h\sum_{j\in A_i}(1 + |X_{t_{j-1}}|^C)\right]\\ &\lesssim \max_{k\in\{2,3\}} \sup_{\gamma\in\Theta_\gamma} E\left[\left|\sum_{j\in A_i} \partial_\gamma^k (c^{-1})_{j-1}(\gamma) \int_{t_{j-1}}^{t_j} c_{s-} dJ_s\right|^r\right] + 1.\end{aligned}$$

Let $\chi_j(s)$ denote the indicator function of the interval $[t_{j-1}, t_j)$.

To proceed, we recall Burkholder's inequality for stochastic integrals with respect to a centered Lévy process: under the moment conditions on J_η, for any predictable process H and $q \geq 2$ we have

$$\begin{aligned}&E\left[\left|\int_{\lfloor \frac{i-1}{h}\rfloor h}^{\lfloor \frac{i}{h}\rfloor h} H_s dJ_s\right|^q\right]\\ &\leq K_q(\nu_{\eta,2}^{q/2} + \nu_{\eta,q}) \int_{\lfloor \frac{i-1}{h}\rfloor h}^{\lfloor \frac{i}{h}\rfloor h} E\left[|H_s|^q\right] ds \lesssim \int_{\lfloor \frac{i-1}{h}\rfloor h}^{\lfloor \frac{i}{h}\rfloor h} E\left[|H_s|^q\right] ds,\end{aligned}$$

where K_q is a positive constant depending only on q, and $\nu_{\eta,k} := \int z^k \nu_\eta(dz)$ for $k \geq 2$ (see [14, Theorem IV 48]).

Then, we see that the last expectation equals

$$E\left[\left|\int_{\lfloor\frac{i-1}{h}\rfloor h}^{\lfloor\frac{i}{h}\rfloor h}\sum_{j\in A_i}\chi_j(s)\partial_\gamma^k(c^{-1})_{j-1}(\gamma)c_{s-}dJ_s\right|^r\right]$$
$$\lesssim\sum_{j\in A_i}\int_{t_{j-1}}^{t_j}E\left[\left|\partial_\gamma^k(c^{-1})_{j-1}(\gamma)\,c_s\right|^r\right]ds\lesssim 1. \tag{18}$$

This together with (16) concludes (17).

Turning to $\zeta_{2,i}$, we note the standard moment estimate: for any real $r\geq 2$,

$$\max_{j\leq n}\sup_{s\in(t_{j-1},t_j]}E\left[|X_s-X_{t_{j-1}}|^r\right]\lesssim h.$$

With this and the Lipschitz property of $x\mapsto(a(x,\alpha_0),c(x,\gamma_0))$, analogous arguments as in handling $\zeta_{1,i}$ yield that for each $r\geq 2$

$$\max_{1\leq i\leq\lfloor T_n\rfloor}E\left[|\zeta_{2,i}|^r\right]\lesssim h. \tag{19}$$

As for the remaining $\zeta_{3,i}$, it follows from Assumption 2.1 and the stationarity of increments that for each $r\geq 2$,

$$\max_{1\leq i\leq\lfloor T_n\rfloor}E\left[|\zeta_{3,i}|^r\right]\lesssim E\left[\left(J_{j-1}-J_{\lfloor\frac{j-1}{h}\rfloor h+h}\right)^r\right]+E\left[\left(J_{\lfloor\frac{j}{h}\rfloor h}-J_j\right)^r\right]\lesssim h. \tag{20}$$

Piecing together (17), (19), and (20) concludes the proof. □

Proof of Theorem 3.1. Mimicking the estimates for (17), for each $r\geq 2$ we obtain

$$\max_{1\leq i\leq\lfloor T_n\rfloor}E\left[\left|\frac{1}{\sqrt{T_n}}b_i\right|^r\right]\lesssim T_n^{-r/2}.$$

Combined with (15) and (16), it follows from Hölder's inequality that for each $r\geq 2$,

$$\max_{1\leq i\leq\lfloor T_n\rfloor}E\left[|\hat\epsilon_i-\epsilon_i|^r\right]\lesssim h\vee T_n^{-\frac{r}{2}}, \tag{21}$$

and hence $\max_{1\leq i\leq\lfloor T_n\rfloor}E[|\hat\epsilon_i|^r]\lesssim 1$ as well. We use the expression

$$\sum_{i=1}^{\lfloor T_n\rfloor}\rho\left(\hat\epsilon_i\right)=\sum_{i=1}^{\lfloor T_n\rfloor}\rho\left(\epsilon_i\right)+\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon\rho\left(\epsilon_i\right)\left(\hat\epsilon_i-\epsilon_i\right)$$
$$+\frac{1}{2}\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon^2\rho\left(\epsilon_i+u\left(\hat\epsilon_i-\epsilon_i\right)\right)\left(\hat\epsilon_i-\epsilon_i\right)^2$$

for a (random) $u\in[0,1]$. By means of Schwarz's inequality and (15),

$$E\left[\left|\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon\rho\left(\epsilon_i\right)\left(\hat\epsilon_i-\epsilon_i\right)-\frac{1}{\sqrt{T_n}}\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon\rho\left(\epsilon_i\right)b_i[\hat u_{\theta,n}]\right|\right]$$

$$= E\left[\left|\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon\rho\left(\epsilon_i\right)\left(\hat{\epsilon}_i-\epsilon_i-\frac{1}{\sqrt{T_n}}b_i[\hat{u}_{\theta,n}]\right)\right|\right]\lesssim T_n\sqrt{h\vee T_n^{-2}} \tag{22}$$
$$= \sqrt{T_n}\sqrt{nh^2\vee T_n^{-1}}=o\left(\sqrt{T_n}\right).$$

By the moment estimates in the proof of Lemma 3.2, Hölder's inequality, and (21), and also recalling (2), we see that for $\delta\in(1,2]$,

$$E\left[\left|\sum_{i=1}^{\lfloor T_n\rfloor}\partial_\epsilon^2\rho\left(\epsilon_i+u\left(\hat{\epsilon}_i-\epsilon_i\right)\right)\left(\hat{\epsilon}_i-\epsilon_i\right)^2\right|\right]$$
$$\lesssim T_n\max_{1\le i\le\lfloor T_n\rfloor}E\left[\left|\hat{\epsilon}_i-\epsilon_i\right|^{2\delta}\right]^{1/\delta}\lesssim\sqrt{T_n}\left\{(nh^{1+2/\delta})^{\delta/2}\vee T_n^{-\delta/2}\right\}^{1/\delta}=o\left(\sqrt{T_n}\right).$$

This completes the proof. □

3.2. M-estimation of noise parameter

We keep Assumptions 2.1 to 2.5 in force. Having Theorem 3.1 in hand, we proceed with estimation of η based on the unit-time residual sequence $(\hat{\epsilon}_i)$. Let

$$\mathbb{H}_{3,n}(\eta):=\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n\rfloor}m(\hat{\epsilon}_i,\eta),$$

and consider an M-estimator

$$\hat{\eta}_n\in\operatorname*{argmax}_{\eta\in\overline{\Theta_\eta}}\mathbb{H}_{3,n}(\eta). \tag{23}$$

Among others, this includes the (quasi) maximum-likelihood for $m(\epsilon,\eta)=:\log f(\epsilon;\eta)$, where $\{f(\epsilon;\eta):\eta\in\Theta_\eta\}$ is a model for the unit-time noise distribution $\mathcal{L}(J_1)$. We need to impose several conditions on the function m, all of which are standard in the general theory of M-estimation.

Assumption 3.3.

1. *$m\in\mathcal{C}^{2,3}(\mathbb{R}\times\Theta_\eta)$ and $\max_{\substack{i\in\{0,1,2\}\\k\in\{0,1,2,3\}}}\sup_{\eta\in\overline{\Theta_\eta}}\left|\partial_\epsilon^i\partial_\eta^k m(\epsilon,\eta)\right|\lesssim 1+|\epsilon|^C$ for some $C\ge 0$.*
2. *$E\left[\partial_\eta m(J_1,\eta_0)\right]=0$, the $p_\eta\times p_\eta$-matrix $E\left[\left(\partial_\eta m(J_1,\eta_0)\right)^{\otimes 2}\right]$ is positive definite, and*
$$\{\eta_0\}=\operatorname*{argmax}_\eta E\left[m(J_1,\eta)\right].$$

The consistency of $\hat{\eta}_n$ can be easily seen from Theorem 3.1 and Lemma 3.2: we have the continuous random function

$$\mathbb{Y}_n(\eta):=\mathbb{H}_{3,n}(\eta)-\mathbb{H}_{3,n}(\eta_0)\xrightarrow{p}\mathbb{Y}(\eta):=E\left[m(J_1,\eta)\right]-E\left[m(J_1,\eta_0)\right],$$

where $\mathbb{Y}(\eta) \leq 0$ by Jensen's inequality with $\mathbb{Y}(\eta) = 0$ if and only if $\eta = \eta_0$, and moreover, the convergence is uniform in $\eta \in \overline{\Theta_\eta}$ since $\sup_n E[\sup_\eta |\partial_\eta \mathbb{Y}_n(\eta)|] < \infty$. Hence the consistency $\hat{\eta}_n \xrightarrow{p} \eta_0$ follows.

We turn to the asymptotic normality of $\hat{\eta}_n$. Let $\hat{u}_{\eta,n} := \sqrt{T_n}(\hat{\eta}_n - \eta_0)$. In the sequel, for any measurable function $f(x,\theta)$ we will simply write $\hat{f}_{j-1}$ for $f_{j-1}(\hat{\theta}_n)$.

Corollary 3.4. *Under Assumptions 2.1 to 2.5, and Assumption 3.3, we have*

$$\begin{aligned}&\left(-\partial_\eta^2 \mathbb{H}_{3,n}(\hat{\eta}_n)\right)[\hat{u}_{\eta,n}] - \frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}\left(\partial_\eta \partial_\epsilon m\left(\hat{\epsilon}_i, \hat{\eta}_n\right)\left(\hat{b}_i[\hat{u}_{\theta,n}]\right)\right)\\ &\xrightarrow{\mathcal{L}} N_{p_\eta}\left(0, E\left[\left(\partial_\eta m(J_1,\eta_0)\right)^{\otimes 2}\right]\right),\end{aligned}$$

where

$$\hat{b}_i := \sum_{j \in A_i}\begin{pmatrix}\widehat{\partial_\gamma (c^{-1})}_{j-1}(\Delta_j X - h\hat{a}_{j-1})\\ -h\widehat{c^{-1}}_{j-1}\widehat{\partial_\alpha a}_{j-1}\end{pmatrix}.$$

Proof. By the consistency $\hat{\eta}_n \xrightarrow{p} \eta_0$ we may and do focus on the event $\{\partial_\eta \mathbb{H}_{3,n}(\hat{\eta}_n) = 0\}$, on which

$$\left(-\int_0^1 \partial_\eta^2 \mathbb{H}_{3,n}(\eta_0 + s(\hat{\eta}_n - \eta_0))ds\right)[\hat{u}_{\eta,n}] = \sqrt{T_n}\partial_\eta \mathbb{H}_{3,n}.$$

Then, by Theorem 3.1 we have

$$\begin{aligned}\left(-\partial_\eta^2 \mathbb{H}_{3,n}(\hat{\eta}_n) + o_p(1)\right)[\hat{u}_{\eta,n}] &= \frac{1}{\sqrt{T_n}}\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta m\left(\epsilon_i, \eta_0\right)\\ &\quad + \frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}\left(\partial_\eta \partial_\epsilon m\left(\hat{\epsilon}_i, \hat{\eta}_n\right)\left(\hat{b}_i[\hat{u}_{\theta,n}]\right)\right) + o_p(1).\end{aligned}$$

Hence, letting

$$\hat{u}_n := \left(\hat{u}_{\theta,n}, \hat{u}_{\eta,n}\right),$$

we have

$$\hat{H}_n[\hat{u}_n] = \frac{1}{\sqrt{T_n}}\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta m\left(\epsilon_i, \eta_0\right) + o_p(1), \tag{24}$$

where $\hat{H}_n$ is $p_\eta \times (p + p_\eta)$-matrix given by

$$\hat{H}_n = \left(-\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta \partial_\epsilon m\left(\hat{\epsilon}_i, \hat{\eta}_n\right)\hat{b}_i, \quad -\partial_\eta^2 \mathbb{H}_{3,n}(\hat{\eta}_n)\right),$$

and is $O_p(1)$. Since the random variables $\epsilon_1, \epsilon_2, \ldots, \epsilon_{\lfloor T_n \rfloor}$ are i.i.d., the desired result follows on applying the central limit theorem to (24). □

Corollary 3.4 suggests that the effect of plugging in the GQMLE does remain in the limit. Therefore, in order to construct confidence interval and hypothesis testing for η, we need to verify the joint asymptotic distribution of $\hat{\mathcal{I}}_n[\hat{u}_n]$ with some invertible matrix $\hat{\mathcal{I}}_n$. We additionally introduce the following condition.

Assumption 3.5. *There exist $p_\eta \times p_\gamma$-matrix $\Sigma_{\eta,\gamma}$ and $p_\eta \times p_\alpha$-matrix $\Sigma_{\eta,\alpha}$ such that*

$$\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor} \partial_\eta m(\epsilon_i,\eta_0) \sum_{j\in A_i}\left(\frac{\partial_\gamma c_{j-1}}{c_{j-1}}\left[(\Delta_j J)^2 - h\right]\right) \xrightarrow{p} \Sigma_{\eta,\gamma}, \tag{25}$$

$$\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor} \partial_\eta m(\epsilon_i,\eta_0) \sum_{j\in A_i}\left(\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\Delta_j J\right) \xrightarrow{p} \Sigma_{\eta,\alpha}. \tag{26}$$

Furthermore, the $(p+p_\eta)\times(p+p_\eta)$-matrix

$$\Sigma = \begin{pmatrix} \Sigma_\gamma & \Sigma_{\gamma,\alpha} & \Sigma_{\eta,\gamma} \\ \Sigma_{\gamma,\alpha}^\top & \Sigma_\alpha & \Sigma_{\eta,\alpha} \\ \Sigma_{\eta,\gamma}^\top & \Sigma_{\eta,\alpha}^\top & E\left[(\partial_\eta m(J_1,\eta_0))^{\otimes 2}\right] \end{pmatrix},$$

is invertible.

From now on, we will write $E^{i-1}[\cdot]$ for the conditional expectation $E[\cdot|\mathcal{F}_{i-1}]$.

Remark 3.6. *By an elementary application of Burkholder's inequality, it is easy to deduce that (25) and (26) are equivalent to*

$$\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}\left[\partial_\eta m(\epsilon_i,\eta_0) \sum_{j\in A_i}\left(\frac{\partial_\gamma c_{j-1}}{c_{j-1}}\left[(\Delta_j J)^2 - h\right]\right)\right] \xrightarrow{p} \Sigma_{\eta,\gamma},$$

$$\frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}\left[\partial_\eta m(\epsilon_i,\eta_0) \sum_{j\in A_i}\left(\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\Delta_j J\right)\right] \xrightarrow{p} \Sigma_{\eta,\alpha},$$

respectively. This fact will be used later.

Let

$$\hat{\Gamma}_n := \mathrm{diag}(-\partial_\gamma^2 \mathbb{H}_n(\hat{\gamma}_n), -\partial_\alpha^2 \mathbb{H}_n(\hat{\alpha}_n)),$$

$$\hat{\mathcal{I}}_n := \begin{pmatrix} \hat{\Gamma}_n & O \\ -T_n^{-1}\sum_{i=1}^{\lfloor T_n \rfloor} \partial_\eta \partial_\epsilon m\left(\hat{\epsilon}_i,\hat{\eta}_n\right)\hat{b}_i & -\partial_\eta^2 \mathbb{H}_{3,n}(\hat{\eta}_n) \end{pmatrix}.$$

We also introduce the $(p+p_\eta)\times(p+p_\eta)$-matrix

$$\hat{\Sigma}_n = \begin{pmatrix} \hat{\Sigma}_{\gamma,n} & \hat{\Sigma}_{\gamma,\alpha,n} & \hat{\Sigma}_{\eta,\gamma,n} \\ \hat{\Sigma}_{\gamma,\alpha,n}^\top & \hat{\Sigma}_{\alpha,n} & \hat{\Sigma}_{\eta,\alpha,n} \\ \hat{\Sigma}_{\eta,\gamma,n}^\top & \hat{\Sigma}_{\eta,\alpha,n}^\top & \hat{\Sigma}_{\eta,n} \end{pmatrix},$$

where the ingredients are defined as follows:

$$\hat{\Sigma}_{\gamma,n} = \left(\frac{1}{n}\sum_{j=1}^{n}\left(\frac{\partial_\gamma \hat{c}_{j-1}}{\hat{c}_{j-1}}\right)^{\otimes 2}\right)\left(\frac{1}{h_n}\sum_{j=1}^{n}\left(\widehat{\Delta_j J}\right)^4\right),$$

$$\hat{\Sigma}_{\alpha,n} = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{\partial_\alpha \hat{a}_{j-1}}{\hat{c}_{j-1}}\right)^{\otimes 2},$$

$$\hat{\Sigma}_{\eta,n} = \frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}(\partial_\eta m(\hat{\epsilon}_i, \hat{\eta}_n))^{\otimes 2},$$

$$\hat{\Sigma}_{\gamma,\alpha,n} = \left(\frac{1}{n}\sum_{j=1}^{n}\frac{\partial_\gamma \hat{c}_{j-1}\partial_\alpha \hat{a}_{j-1}}{\hat{c}_{j-1}^2}\right)\left(\frac{1}{h_n}\sum_{j=1}^{n}\left(\widehat{\Delta_j J}\right)^3\right),$$

$$\hat{\Sigma}_{\eta,\gamma,n} = \frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta m(\hat{\epsilon}_i, \hat{\eta}_n)\sum_{j\in A_i}\left(\frac{\partial_\gamma \hat{c}_{j-1}}{\hat{c}_{j-1}}\left[\left(\widehat{\Delta_j J}\right)^2 - h\right]\right),$$

$$\hat{\Sigma}_{\eta,\alpha,n} = \frac{1}{T_n}\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta m(\hat{\epsilon}_i, \hat{\eta}_n)\sum_{j\in A_i}\left(\frac{\partial_\alpha \hat{a}_{j-1}}{\hat{c}_{j-1}}\widehat{\Delta_j J}\right).$$

Now we are ready to state the main result.

Theorem 3.7. *Under Assumptions 2.1 to 2.5, Assumption 3.3, and Assumption 3.5, we have*

$$\hat{\Sigma}_n^{-1/2}\hat{\mathcal{I}}_n[\hat{u}_n] \xrightarrow{\mathcal{L}} N_{p+p_\eta}\left(0, I_{p+p_\eta}\right).$$

By Theorem 3.7, we have

$$T_n(\hat{\theta}_n - \theta_0, \hat{\eta}_n - \eta_0)\,\hat{\mathcal{I}}_n^\top \hat{\Sigma}_n^{-1}\hat{\mathcal{I}}_n(\hat{\theta}_n - \theta_0, \hat{\eta}_n - \eta_0)^\top \xrightarrow{\mathcal{L}} \chi^2(p + p_\eta),$$

based on which we can construct an approximate confidence set for (θ_0, η_0), and also perform a Wald-type test. Also trivially, we can recover the asymptotic distribution of $\hat{\theta}_n$:

$$T_n(\hat{\theta}_n - \theta_0)\,\hat{\Gamma}_n\begin{pmatrix}\hat{\Sigma}_{\gamma,n} & \hat{\Sigma}_{\gamma,\alpha,n}\\ \hat{\Sigma}_{\gamma,\alpha,n}^\top & \hat{\Sigma}_{\alpha,n}\end{pmatrix}^{-1}\hat{\Gamma}_n(\hat{\theta}_n - \theta_0)^\top \xrightarrow{\mathcal{L}} \chi^2(p).$$

It is difficult to obtain $\Sigma_{\eta,\gamma}$ and $\Sigma_{\eta,\alpha}$ in explicit easy-to-handle forms even if the coefficients and m are simple. However, by an application of Cauchy-Schwartz inequality and the estimates in the proof of Theorem 3.8, we can observe that at least, the left-hand-sides in (25) and (26) are tight. Moreover, we can formally write their limit by means of the representation theorem ([10, Proposition 3]): there exists a predictable process $s \mapsto \tilde{\xi}_{\eta,n}(s,z)$ such that

$$\sum_{i=1}^{\lfloor T_n \rfloor}\partial_\eta m(\epsilon_i, \eta_0) = \int_0^{\lfloor T_n \rfloor}\int \tilde{\xi}_{\eta,n}(s,z)\tilde{N}(ds,dz). \tag{27}$$

From Itô's formula and some calculations, we have

$$
\begin{aligned}
&\sum_{j\in A_i} \frac{\partial_\alpha a_{j-1}}{c_{j-1}} \Delta_j J \\
&= \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \sum_{j\in A_i} \left(\chi_j(s) \frac{\partial_\alpha a_{j-1}}{c_{j-1}} \right) dJ_s \\
&= \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \int \sum_{j\in A_i} \left(\chi_j(s) \frac{\partial_\alpha a_{j-1}}{c_{j-1}} \right) z\tilde{N}(ds,dz) \\
&= \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \int \frac{\partial_\alpha a_{s-}}{c_{s-}} z\tilde{N}(ds,dz), \\
&\sum_{j\in A_i} \frac{\partial_\gamma c_{j-1}}{c_{j-1}} \left[(\Delta_j J)^2 - h \right] \\
&= \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \sum_{j\in A_i} \left(\chi_j(s) \frac{\partial_\gamma c_{j-1}}{c_{j-1}} (J_{s-} - J_{t_{j-1}}) \right) dJ_s \\
&+ \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \int \sum_{j\in A_i} \left(\chi_j(s) \frac{\partial_\gamma c_{j-1}}{c_{j-1}} \right) z^2 \tilde{N}(ds,dz) \\
&= \int_{\lfloor \frac{i-1}{h} \rfloor h}^{\lfloor \frac{i}{h} \rfloor h} \int \frac{\partial_\gamma c_{s-}}{c_{s-}} z^2 \tilde{N}(ds,dz) + O_p(h_n).
\end{aligned}
$$

To sum up, we obtain the following expression:

$$
\hat{\mathcal{I}}_n[\hat{u}_n] = \frac{1}{\sqrt{T_n}} \int_0^{\lfloor T_n \rfloor} \int \begin{pmatrix} \frac{\partial_\gamma c_{s-}}{c_{s-}} z^2 \\ \frac{\partial_\alpha a_{s-}}{c_{s-}} z \\ \tilde{\xi}_{\eta,n}(s,z) \end{pmatrix} \tilde{N}(ds,dz) + o_p(1).
$$

By applying the central limit theorem for the stochastic integral with respect to a Poisson random measure (cf. [19, Lemma A.2]), the isometry property of the stochastic integral yield that under suitable moment and regularity conditions,

$$
\hat{\mathcal{I}}_n[\hat{u}_n] \xrightarrow{\mathcal{L}} N_{p+p_\eta} \left(0, \begin{pmatrix} \Sigma_\gamma & \Sigma_{\gamma,\alpha} & \Sigma_{\eta,\gamma} \\ \Sigma_{\gamma,\alpha}^\top & \Sigma_\alpha & \Sigma_{\eta,\alpha} \\ \Sigma_{\eta,\gamma}^\top & \Sigma_{\eta,\alpha}^\top & E\left[(\partial_\eta m(J_1,\eta_0))^{\otimes 2} \right] \end{pmatrix} \right),
$$

where $\Sigma_{\eta,\gamma}$ and $\Sigma_{\eta,\alpha}$ are given by the limits in probability:

$$
\begin{aligned}
&\frac{1}{T_n} \int_0^{\lfloor T_n \rfloor} \int E\left[\frac{\partial_\gamma c_{s-}}{c_{s-}} \tilde{\xi}_{\eta,n}(s,z) \right] z^2 \nu_0(dz) ds \xrightarrow{p} \Sigma_{\eta,\gamma}, \\
&\frac{1}{T_n} \int_0^{\lfloor T_n \rfloor} \int E\left[\frac{\partial_\alpha a_{s-}}{c_{s-}} \tilde{\xi}_{\eta,n}(s,z) \right] z \nu_0(dz) ds \xrightarrow{p} \Sigma_{\eta,\alpha},
\end{aligned}
$$

and the other ingredients are the same as our previous works (cf. Theorem 2.8). However, since the explicit form of $\tilde{\xi}_{\eta,n}(s,z)$ cannot be obtained in general, it is difficult to check the above convergence.

Finally, we would like to add that Theorem 3.7 and the resulting Wald-type test are valid without Assumption 3.5 if the minimum eigenvalue of $\hat{\Sigma}$ is positive uniformly in n. Such a condition for eigenvalues is often assumed in the context of (non-)linear regression.

Proof of Theorem 3.7. By the Cramér-Wold device, we may and do assume that $p_\gamma = p_\alpha = p_\eta = 1$ without loss of generality. It is straightforward to deduce that $\hat{\Sigma}_n \xrightarrow{p} \Sigma$ from Theorem 2.8, Theorem 3.1, Lemma 3.2, and the estimates we have seen in the previous proofs. Hence, by means of Slutsky's theorem, it suffices to show that

$$\hat{\mathcal{I}}_n[\hat{u}_n] \xrightarrow{\mathcal{L}} N_{p+p_\eta}(0,\Sigma). \tag{28}$$

From [18, Proof of Theorem 3.4], we have

$$\hat{\Gamma}_n[\hat{u}_{\theta,n}] = \frac{1}{\sqrt{T_n}} \sum_{i=1}^{\lfloor T_n \rfloor} \sum_{j\in A_i} \begin{pmatrix} \frac{\partial_\gamma c_{j-1}}{c_{j-1}} \left[(\Delta_j J)^2 - h\right] \\ \frac{\partial_\alpha a_{j-1}}{c_{j-1}} \Delta_j J \end{pmatrix} + o_p(1).$$

This together with (24) and the definition of $\hat{\mathcal{I}}_n$ leads to

$$\hat{\mathcal{I}}_n[\hat{u}_n] = \frac{1}{\sqrt{T_n}} \sum_{i=1}^{\lfloor T_n \rfloor} \begin{pmatrix} \xi_{\gamma,i} \\ \xi_{\alpha,i} \\ \xi_{\eta,i} \end{pmatrix} + o_p(1), \tag{29}$$

where

$$\xi_{\gamma,i} := \sum_{j\in A_i} \frac{\partial_\gamma c_{j-1}}{c_{j-1}} \left[(\Delta_j J)^2 - h\right], \quad \xi_{\alpha,i} := \sum_{j\in A_i} \frac{\partial_\alpha a_{j-1}}{c_{j-1}} \Delta_j J,$$
$$\xi_{\eta,i} := \partial_\eta m(\epsilon_i, \eta_0).$$

By (25), (26), and the arguments in the proof of Corollary 3.4, the martingale central limit theorem concludes (28) if we have the following convergences:

$$\left| \frac{1}{\sqrt{T_n}} \sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}[\xi_{\gamma,i}] \right| + \left| \frac{1}{\sqrt{T_n}} \sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}[\xi_{\alpha,i}] \right| \xrightarrow{p} 0, \tag{30}$$

$$\frac{1}{T_n} \sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}\left[\xi_{\gamma,i}^2\right] \xrightarrow{p} \Sigma_\gamma, \tag{31}$$

$$\frac{1}{T_n} \sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}\left[\xi_{\gamma,i}\xi_{\alpha,i}\right] \xrightarrow{p} \Sigma_{\gamma,\alpha}, \tag{32}$$

$$\frac{1}{T_n} \sum_{i=1}^{\lfloor T_n \rfloor} E^{i-1}\left[\xi_{\alpha,i}^2\right] \xrightarrow{p} \Sigma_\alpha, \tag{33}$$

$$\frac{1}{T_n^2}\sum_{i=1}^{\lfloor T_n\rfloor} E^{i-1}\left[|\xi_{\gamma,i}|^4+|\xi_{\alpha,i}|^4+|\xi_{\eta,i}|^4\right]\xrightarrow{p}0. \tag{34}$$

Trivially $\{(\xi_{\gamma,i},\xi_{\alpha,i},\xi_{\eta,i})\}_{i\le\lfloor T_n\rfloor}$ forms a martingale difference array with respect to $(\mathcal{F}_i)$, since we have $\mathcal{F}_{i-1}\subset\mathcal{F}_{t_j}$ for each $j\in A_i$; this immediately ensures (30). By the arguments in Remark 3.6, we can replace (31), (32), and (33) by

$$\frac{1}{T_n}\sum_{j=1}^{n}\left(\frac{\partial_\gamma c_{j-1}}{c_{j-1}}\left[(\Delta_j J)^2-h\right]\right)^2\xrightarrow{p}\Sigma_\gamma,$$

$$\frac{1}{T_n}\sum_{j=1}^{n}\left(\frac{\partial_\gamma c_{j-1}}{c_{j-1}}\left[(\Delta_j J)^2-h\right]\right)\left(\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\Delta_j J\right)\xrightarrow{p}\Sigma_{\gamma,\alpha},$$

$$\frac{1}{T_n}\sum_{j=1}^{n}\left(\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\Delta_j J\right)^2\xrightarrow{p}\Sigma_\alpha,$$

respectively. Noting that $E[|J_{h_n}|^q]=O(h_n)$ for any $q\ge 2$ under Assumption 2.1, we can deduce the last three convergences from [3, Lemma 9] and the ergodic theorem.

It remains to show (34). It follows from Itô's formula and Assumption 2.1 that for any $j\in\{1,\dots,n\}$,

$$(\Delta_j J)^2-h=2\int_{t_{j-1}}^{t_j}(J_{s-}-J_{t_{j-1}})dJ_s+\int_{t_{j-1}}^{t_j}\int z^2\tilde{N}(ds,dz),$$

where $\tilde{N}(ds,dz)$ is the compensated Poisson random measure of J; recall that we are assuming that $E[(\Delta_j J)^2]=(\sigma^2+\int z^2\nu(dz))h=h$. Then, we can rewrite $\xi_{\gamma,i}$ and $\xi_{\alpha,i}$ as

$$\begin{aligned}\xi_{\gamma,i}&=\int_{\lfloor\frac{i-1}{h}\rfloor h}^{\lfloor\frac{i}{h}\rfloor h}\sum_{j\in A_i}\left(\chi_j(s)\frac{\partial_\gamma c_{j-1}}{c_{j-1}}(J_{s-}-J_{t_{j-1}})\right)dJ_s\\&\quad+\int_{\lfloor\frac{i-1}{h}\rfloor h}^{\lfloor\frac{i}{h}\rfloor h}\int\sum_{j\in A_i}\left(\chi_j(s)\frac{\partial_\gamma c_{j-1}}{c_{j-1}}\right)z^2\tilde{N}(ds,dz),\\\xi_{\alpha,i}&=\int_{\lfloor\frac{i-1}{h}\rfloor h}^{\lfloor\frac{i}{h}\rfloor h}\sum_{j\in A_i}\left(\chi_j(s)\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\right)dJ_s\\&=\int_{\lfloor\frac{i-1}{h}\rfloor h}^{\lfloor\frac{i}{h}\rfloor h}\int\sum_{j\in A_i}\left(\chi_j(s)\frac{\partial_\alpha a_{j-1}}{c_{j-1}}\right)z\tilde{N}(ds,dz).\end{aligned}$$

Taking a similar route to the estimate (18), we have

$$\frac{1}{T_n^2}\sum_{i=1}^{\lfloor T_n\rfloor}E^{i-1}\left[|\xi_{\gamma,i}|^4+|\xi_{\alpha,i}|^4\right]\lesssim\frac{1}{T_n^2}\sum_{i=1}^{\lfloor T_n\rfloor}(1+|X_{i-1}|^C)=O_p(T_n^{-1})=o_p(1).$$

Since $T_n^{-2}\sum_{i=1}^{\lfloor T_n\rfloor}E^{i-1}\left[|\xi_{\eta,i}|^4\right]\xrightarrow{p}0$ under Assumptions 2.1 and 3.3, we obtain the desired result. □

3.3. Further remarks

3.3.1. Dimension of the processes

For the asymptotics of the GQMLE, we could consider multivariate X without any essential change [18]; we will conduct related simulations in Section 5.2. Moreover, the estimator of (α, γ) may not be necessarily the GQMLE and could be any measurable mappings $\hat{\theta}_n = \hat{\theta}_n(\boldsymbol{X}_n)$ for which we have an asymptotically linear representation as in (29).

3.3.2. Model selection for $\mathcal{L}(J_1)$

Residual based on information criterion (IC) formulation after estimation should be possible (both AIC and BIC types). We can infer the structure of J as in the i.i.d case, yet should be careful in making possibly necessary corrections stemming from the stochastic expansion (14). For example, for the AIC statistics to be theoretically in effect, among other conditions it is required that the random sequence $(\hat{u}_{\eta,n})_n$ is $L^{2+\delta}(P)$-bounded for some $\delta > 0$. It could be verified by means of the uniform tail-probability estimate for $(\hat{u}_{\eta,n})_n$ through the random function $\mathbb{H}_{3,n}(\eta)$; indeed, we could make use of the same machinery to deduce Theorem 2.8(2).

3.3.3. Setting of noise inference

Although we have set a finite-dimensional η above, we could consider infinite-dimensional η, most generally $\mathcal{L}(J_1)$ itself: once $\{\hat{\epsilon}_j\}$ has been constructed, it is also possible to take into account conventional nonparametric procedures, such as the kernel density estimation, and also goodness-of-fit tests; see Section 5.3 for an illustration.

4. Implementation

In this section, we discuss the new classes and the new methods in the YUIMA R package that gives us the possibility to deal with an SDE driven by a Lévy process completely specified by the user. To construct an object of YUIMA class, that is a mathematical description of an SDE driven by a pure Lévy jump process, three steps are necessary:

1. Definition of an object that contains all the information about the structure of the pure Lévy jump. In this step, the user can specify a random number generator, a density function, a cumulative distribution function, a quantile function, a characteristic function, and the number of components for the underlying Lévy process.
2. Definition of the structure of the SDE where the driving noise is determined from the object constructed in Step 1.

Object of YUIMA class	
`Slots:`	
@data :	Data that can be either real or simulated
@model :	Mathematical description of the model
@sampling :	Structure of the time grid
@characteristic :	Additional info
@functional :	Functional of SDE

FIG 1. *The structure of an object that belongs to the YUIMA class.*

Object of `yuima.law` class	
`Slots:`	
@rng :	Random number generator
@density :	Density function
@cdf :	Cumulative distribution function
@quantile :	Quantile function
@characteristic :	Characteristic function
@param.measure :	Parameter labels
@time.var :	Time label
@dim :	Dimension of the random variable

FIG 2. *The structure of an object that belongs to the* `yuima.law` *class.*

3. Construction of an object that belongs to the YUIMA class whose slots are reported in Figure 1. The slot `model` is filled with the object built in Step 2. This new object can be used to simulate a sample path by overwriting the slot `sampling` with the structure of the time grid. Alternatively, we can use this object to estimate the SDE defined in Step 2. In this case, we can store the observed data in the slot `data`.

4.1. yuima.law: A new class for a mathematical description of the Lévy process

In this section, we describe the structure of a `yuima.law-object` and its constructor `setLaw`. The main advantage of this new class is the possibility of connecting YUIMA with any CRAN package that provides functions for a specific random variable. Figure 2 reports the slots that constitute an object of `yuima.law` class.

The first five slots contain R user-defined functions. In particular, the first two slots contain the random number generator and the density function respectively. Although it is not necessary to specify these functions to construct an object of `yuima.law` class, the definition of a random number generator is necessary to run the YUIMA `simulate` method while the density function is

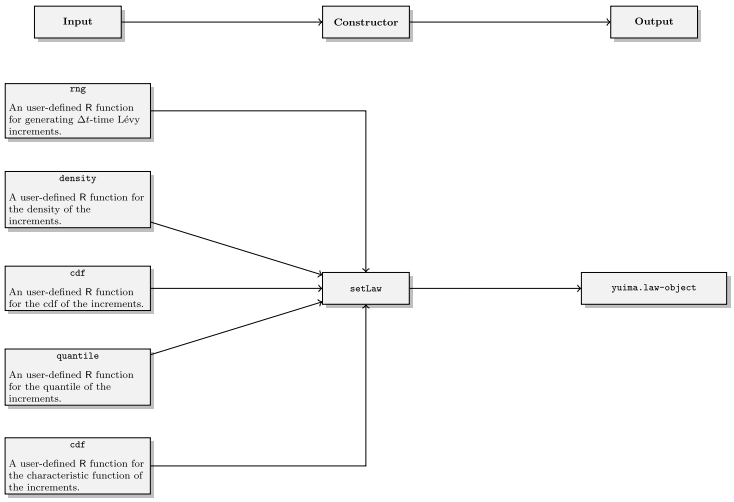

FIG 3. *Procedure for the construction of an object that belongs to the* `yuima.law` *class.*

used internally by the YUIMA `qmleLevy` method. The template of these two functions is listed below:

```
# User specified random number generator
R> user.rng <- function(n, eta, t){
+    ... ... ... # Body of the function
+  }
# User specified density function
R> user.density <- function(x, eta, t){
+    ... ... ... # Body of the function
+  }
```

where the input `eta` is a `vector` containing the names of the Lévy noise parameters and the input `t` refers to the label of the time variable.

An object of `yuima.law` class is built using `setLaw` constructor.

```
R> setLaw(rng = function(n, ...){ NULL },
+    density = function(x, ...){ NULL },
+    cdf = function(q, ...){ NULL },
+    quant = function(p, ...){ NULL },
+    characteristic = function(u, ...){ NULL },
+    time.var = "t", dim = NA )
```

The first five inputs in the function fill the corresponding slots in the `yuima.law` object. Figure 3 describes the steps required for the construction of the `yuima.law-object`.

After the construction of an object that belongs to the `yiuma.law` class, by using the standard constructor `setModel` where an `yuima.law` object is passed

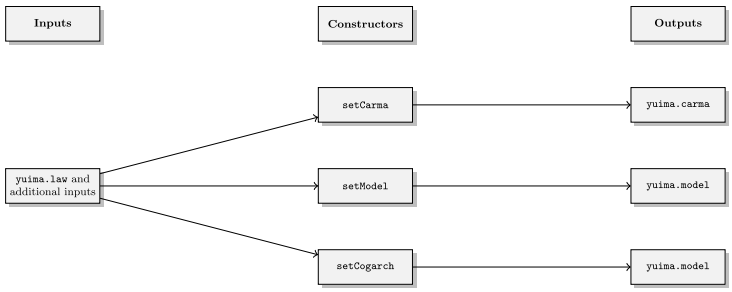

FIG 4. *Procedure for the definition of the model where the noise is defined by an object of* `yuima.law` *class.*

to `setModel` through the argument `measure`, the user can specify completely a SDE driven by a pure Lévy jump as shown in the following command line:

```
R> setModel(drift = "User.Defined_drift",
+    jump.coeff = "User.Defined_jump.coef", measure.type = "code",
+    measure = list(df = User.Defined_yuima.law))
```

We remark that an object of `yuima.law` class can be also used to specify the Lévy noise in the Continuous Time ARMA model [4] and in the COGARCH process [5, 6]. In the first case, the model is built using the constructor `setCarma`:

```
R> setCarma(p, q, measure.type = "code", measure = list(df =
+    User.Defined_yuima.law))
```

where p and q are two integers indicating the order of the autoregressive and the moving average parameters. The COGARCH(p,q) process can be defined in YUIMA using the function `setCogarch` as follows:

```
R> setCogarch(p, q, measure.type = "code", measure = list(df =
+    User.Defined_yuima.law))
```

Figure 4 shows how to use an object of `yuima.law` class in the definition of models that can be constructed using YUIMA.

4.2. yuima.qmleLevy.incr: Estimation of an SDE driven by a Lévy pure jump process in yuima

In this section, we discuss how to estimate an SDE driven by a Lévy pure jump process in YUIMA. In particular, we describe the features of the new class `yuima.qmleLevy.incr` and explain the usage of the new method `qmleLevy`. The `yuima.qmleLevy.incr` class is the extension of the classical `yuima.qmle` class because we have additional slots associated with the filtered Lévy increments obtained using the procedure described in Section 3. As a child class, `yuima.qmleLevy.incr` class inherits all the YUIMA methods developed for the

Object of `yuima.qmleLevy.incr` class	
@Incr.Lev :	Estimated Δt or unit-time Lévy increments
@logL.Incr :	Log-likelihood of the estimated Lévy increments
@Levydetails :	Additional information on the internal optimization
@Data :	Observed data
@... :	Slots inherited from `yuima.qmle` class

FIG 5. *Main slots of an object that belongs to the* `yuima.qmleLevy.incr` *class.*

`yuima.qmle` class. Figure 5 reports the new slots. The most relevant for our study is the slot `Incr.Lev` where we can find the estimated Lévy increments.

An object of `yuima.qmleLevy.incr` class can not be directly constructed by the user but it is a possible output of the function `qmleLevy` that performs the estimation approach discussed in Section 2. The syntax of this function is as follows:

```
R> qmleLevy(yuima, start, lower, upper, joint = FALSE,
+    third = FALSE, Est.Incr = "NoIncr", aggregation = TRUE)
```

The first argument is an object of `yuima` class where the slot `data` contains the observed dataset, while the slot `model` is a mathematical description of the SDE driven by the pure Lévy jump process. The arguments `start`, `lower` and `upper` are used in the optimization routine to identify the initial guesses and box-constraints. The arguments `joint` and `third` are technical arguments related to the procedure of the GQMLE; we refer to [7] for a specific documentation of their meaning. The most important arguments for the estimation of the Lévy increments are `Est.Incr` and `aggregation`. The argument `Est.Incr` assumes three values: `NoIncr`, `Incr` and `IncrPar`. In the first case, the function returns an object of `yuima.qmle` class that contains only the SDE parameters. The function `qmleLevy` internally runs only the GQMLE procedure. Setting `Est.Incr = "Incr"` or `Est.Incr = "IncrPar"`, `qmleLevy` returns an object of `yuima.qmleLevy.incr` class. In the first case the object contains the estimated increments while, in the second case, we also obtain the estimated parameters of the Lévy measure. The last argument `aggregation` is a logical variable. If `aggregation = TRUE`, the estimated Lévy increments are (4) associated to the unit-time intervals while, if `aggregation = FALSE`, the function returns the Δt-time Lévy increments, see (3).

Figure 6 shows all the steps for the estimation of an SDE driven by a user-defined Lévy process using the real data. As remarked at beginning of Section 4, Figure 6 remarks the preliminary step for the construction of an object that belongs to `yuima` class through the constructor `setYuima`.

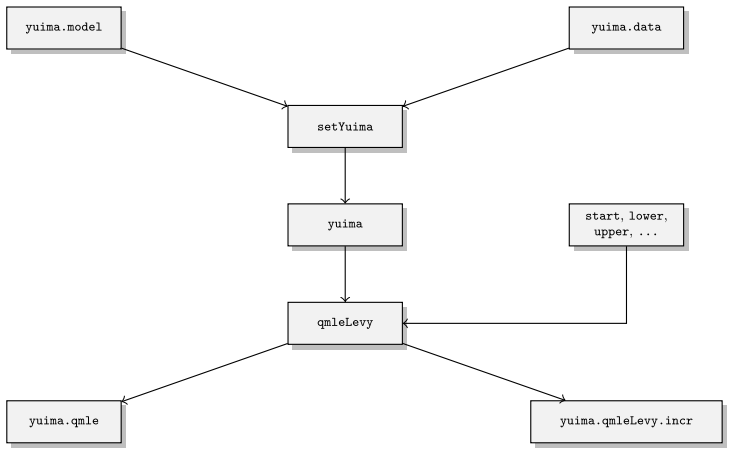

FIG 6. *Procedure for the estimation of a SDE driven by a Lévy process in YUIMA.*

5. Numerical examples

5.1. Univariate Lévy SDE model

In this section, we show how to use YUIMA in the simulation and estimation of an univariate SDE driven by a pure jump Lévy process defined by the user through an object of `yuima.law` class. The model, that we consider, is defined by the following SDE:

$$dX_t = \alpha_1 (\alpha_2 - X_t)\, dt + \gamma \mathrm{d}J_t \tag{35}$$

where α_2 is a real parameter while α_1 and γ are positive parameters, and where $\{J_t\}_{t\geq 0}$ is a symmetric Variance Gamma process with parameter $\eta > 0$.

In this example we use an object of `yuima.law` class to construct a link between YUIMA and the `VarianceGamma` package [16] available in CRAN. We use two functions available in the package `VaranceGamma` respectively `rvg` for the random number generation and `dvg` to construct the density function. The parametrization in `VarianceGamma` package was introduced in [11] where the symmetric Variance Gamma J random variable is defined as a normal variance mean mixture with a gamma subordinator. Specifically, we set

$$\phi_{J_1}(u) = \left(1 + \sigma^2 \nu \frac{u^2}{2}\right)^{-\frac{1}{\nu}}$$

for the characteristic function of J_1. Setting $\nu = \frac{1}{\eta \Delta t}$ and $\sigma = \sqrt{\Delta t}$, we identify the distribution of the increments $J_t - J_{t-\Delta t}$ for the symmetric Variance Gamma Lévy process used in (35).

Following the structure presented in Section 4, we define an object of `yuima.law` class that contains all the information on the underlying process $\{J_t\}_{t\geq 0}$. We run all examples using version `yuima.1.15.4` available on R-Forge.

```
R> library(VarianceGamma)

#### Definition of a yuima.law object ####

R> myrng <- function(n, eta, t){
+    rvg(n, vgC = 0, sigma = sqrt(t), theta = 0, nu = 1/(eta*t))
+  }

R> mydens <- function(x, eta, t){
+    dvg(x, vgC = 0, sigma = sqrt(t), theta = 0, nu = 1/(eta*t))
+  }

R> mylaw <- setLaw(rng = myrng, density = mydens, dim = 1)

R> class(mylaw)

[1] "yuima.law"
attr(,"package")
[1] "yuima"

R> slotNames(mylaw)

[1] "rng"            "density"        "cdf"            "quantile"
[5] "characteristic" "param.measure"  "time.var"       "dim"
```

Using the constructor `setLaw` we are able to build an object of `yuima.law` class where the first two slots contain the random number generator (`myrng`) and the density function (`mydens`) that we will use for the simulation and the estimation of the distribution of J in the model (35). The next step is to build an object of `yuima.model` class using the standard constructor `setModel`:

```
#### Definition of an object of yuima.model class ####

R> yuima1 <- setModel(drift = "alpha1*(alpha2-X)",
+    jump.coeff = "gamma", jump.variable = "J",
+    solve.variable = c("X"), state.variable = c("X"),
+    measure.type = "code", measure = list(df = mylaw))
```

It is worth noticing that the slot `measure` of the object `yuima1` contains the object `mylaw` constructed previously.

```
R> print(yuima1@measure[[1]])

An object of class "yuima.law"
Slot "rng":
function(n, eta, t){
  rvg(n, vgC = 0, sigma = sqrt(t), theta = 0, nu = 1/(eta*t))
}

Slot "density":
```

```
function(x, eta, t){
  dvg(x, vgC = 0, sigma = sqrt(t), theta = 0, nu = 1/(eta*t))
}

Slot "cdf":
function(q,...){NULL}
<environment: 0x000001e202715cf0>

Slot "quantile":
function(p,...){NULL}
<environment: 0x000001e202715cf0>

Slot "characteristic":
function(u,...){NULL}
<environment: 0x000001e202715cf0>

Slot "param.measure":
[1] "eta"

Slot "time.var":
[1] "t"

Slot "dim":
[1] NA
```

We can generate a sample path using the `simulate` method in YUIMA that we report in Figure 7. The simulation scheme in YUIMA is based on the Euler discretization, and the small-time increments of the noise J therein are generated by the random number generator stored in `mylaw` object.

```
#### real parameters ####
R> alpha1 <- 0.4; alpha2 <- 0.25; gamma<- 0.25; eta <- 1

#### Sample grid ####
R> n <- 50000
R> Time <- 1000
R> sam <- setSampling(Terminal = Time, n = n)

#### Simulation ####
R> yuima2 <- setYuima(model = yuima1, sampling = sam)
R> true <- list(alpha1 = alpha1, alpha2 = alpha2,
+    gamma = gamma, eta = eta)
R> set.seed(123)
R> yuima3 <- simulate(yuima2, true.parameter = true,
+    sampling = sam)

#### plot sample path ####
```

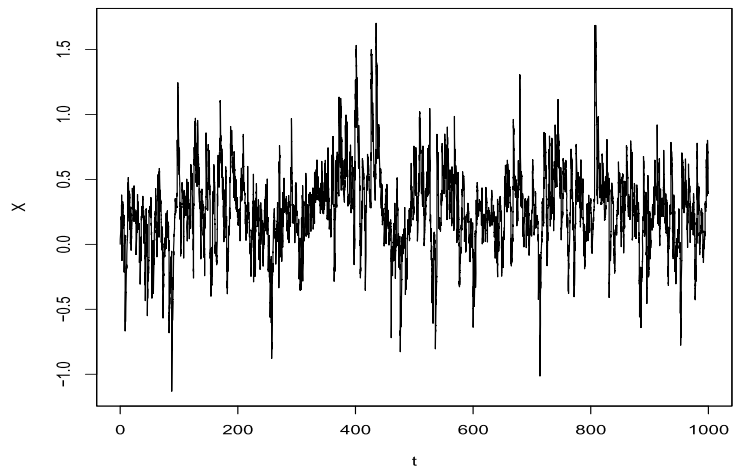

FIG 7. *Simulated Trajectory of an OU-VG model with parameters* $\alpha_1 = 0.4, \alpha_2 = 0.25, \gamma = 0.25, \eta = 1$.

```
R> plot(yuima3)
```

To assess numerically the effectiveness of the three-step estimation procedure discussed in Section 2 and Section 3 we re-estimate the model in (35) using the data stored in the object `yuima3`.

```
#### starting point ####
R> set.seed(123)
R> start <- list(alpha1 = runif(1, 0.01, 2),
+    alpha2 = runif(1, 0.01, 2), gamma = runif(1, 0.01, 2),
+    eta = runif(1, 0.5, 1.5))

#### upper and lower bounds ####
R> upper <- list(alpha1 = 2, alpha2 = 2, gamma = 2, eta = 1.5)
R> lower <- list(alpha1 = 0.01, alpha2 = 0.01, gamma = 0.01,
+    eta = .5)

#### GQMLE procedure ####
R> res.VG <- qmleLevy(yuima3, start = start, lower = lower,
+    upper = upper, Est.Incr = "IncrPar", aggregation = TRUE,
+    joint = FALSE)
```

The function `qmleLevy` returns an object of `yuima.qmleLevy.incr` class that extends the standard class `yuima.qmle`.

```
R> class(res.VG)
```

```
[1] "yuima.qmleLevy.incr"
attr(,"package")
```

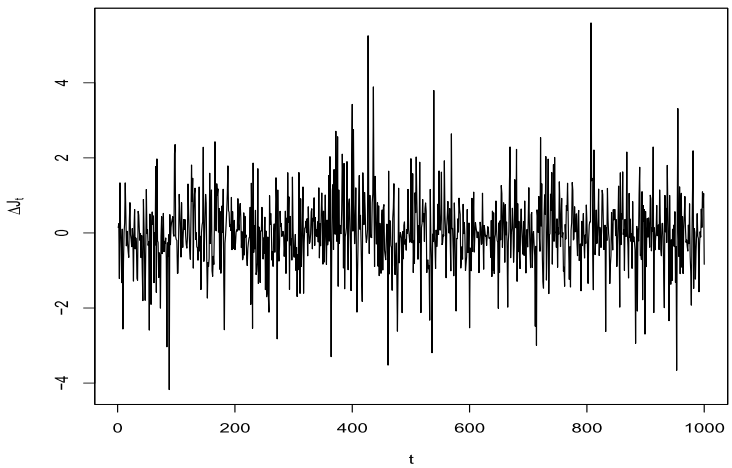

FIG 8. *Estimated unit-time increments from the OU-VG model defined in* (35).

```
[1] "yuima"
```

```
R> slotNames(res.VG)
```

```
 [1] "Incr.Lev"      "logL.Incr"     "minusloglLevy" "Levydetails"
 [5] "Data"          "model"         "call"          "coef"
 [9] "fullcoef"      "fixed"         "vcov"          "min"
[13] "details"       "minuslogl"     "nobs"          "method"
```

The slot `Incr.Lev` is filled with an object of `yuima.data` class that contains the estimated unit-time Lévy increments.

```
R> str(res.VG@Incr.Lev, 2)
```

```
Formal class 'yuima.data' [package "yuima"] with 2 slots
  ..@ original.data:'zooreg' series from 1 to 1000
  Data: num [1:1000, 1] 0.141 0.249 -1.219 1.336 0.106 ...
  .. ..- attr(*, "dimnames")=List of 2
  Index:  num [1:1000] 1 2 3 4 5 6 7 8 9 10 ...
  Frequency: 1
  ..@ zoo.data     :List of 1
```

Figure 8 reports the trajectory of the estimated unit-time Lévy increments.

```
#### Visualization of the estimated unit-time increments ####
R> plot(res.VG@Incr.Lev)
```

5.2. Multivariate Lévy SDE model

In this section, we simulate and estimate a bivariate SDE model driven by two independent symmetric Variance Gamma processes. As done in the previous

section we construct the random number generator and the joint density function of the underlying bivariate Lévy process using the function developed in the `VarianceGamma` package. We report below the code for simulating and estimating the process $X_t := [X_{1,t}, X_{2,t}]^\top$ that satisfies the following system of SDEs:

$$\begin{aligned} dX_{1,t} &= \alpha_{1,1}\left(\alpha_{1,2} - X_{1,t} - 0.2X_{2,t}\right)dt + \gamma_1 dJ_{1,t} \\ dX_{2,t} &= \alpha_{2,1}\left(\alpha_{2,2} - X_{2,t}\right)dt + \gamma_2 dJ_{2,t} \end{aligned} \tag{36}$$

where $J_t = [J_{t,1}, J_{t,2}]^\top$ is a bivariate Lévy process where the components are two independent symmetric Variance Gamma processes.
The first step is to construct an object of `yuima.law` class that contains a random number generator and the joint density of the bivariate Lévy process J_t. As done for the model in (35) we use the functions available in the R package `VarianceGamma`. The random number generator of the increments can be defined using the following command lines:

```
#### Construction of a bivariate rng function ####
R> myrng2 <- function(n, eta1, eta2, t){
+    res0 <- rvg(n, vgC = 0, sigma = sqrt(t), theta = 0,
+            nu = 1 / (eta1 * t))
+    cbind(res0, rvg(n, vgC = 0, sigma = sqrt(t), theta = 0,
+          nu = 1 / (eta2 * t)))
+  }
```

Compared with the random number generator used in the univariate case, the result of the function `rng` is a two-column matrix where each column contains increments generated from a symmetric Variance Gamma random variable. Exploiting the independence assumption we construct the joint density of the process J_t as a product of two univariate symmetric Variance Gamma densities using the following R function:

```
#### Construction of the joint density ####
R> mydens2 <- function(x, eta1, eta2, t){
+    dvg(x[,1], vgC = 0, sigma = sqrt(t), theta = 0,
+        nu = 1/(eta1 * t)) * dvg(x[,2], vgC = 0,
+        sigma = sqrt(t), theta = 0, nu = 1/(eta2 * t))
+  }
```

Using the constructor `setLaw`, we build an object of `yuima.law` that contains information for simulating the noise J_t and for estimating the parameters in (36).

```
R> mylaw2 <- setLaw(rng = myrng2, density = mydens2, dim = 2)
```

We simulate a trajectory of the model in (36) using the standard syntax in YUIMA as follows:

```
#### Model Definition ####
R> yuima2 <- setModel(drift = c("alpha11*(alpha12-X1-0.2*X2)",
+    "alpha21*(alpha22-X2)"), jump.coeff = matrix(c("gamma1",
```

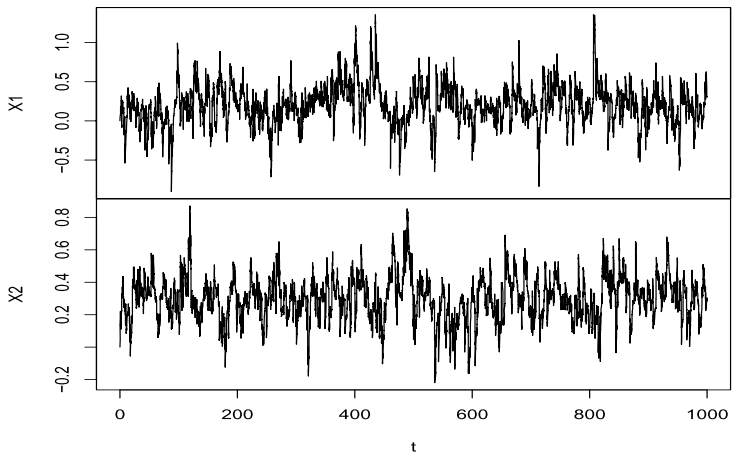

FIG 9. *Simulated trajectory of the bivariate process Y_t defined in* (36).

```
+    0, 0, ""gamma2"), 2, 2), solve.variable = c("X1", "X2"),
+    state.variable = c("X1", "X2"), measure.type = c("code",
+    "code"), measure = list(df = mylaw2), jump.variable = "J")

#### Choosing model parameters ####
R> alpha11 = 0.4; alpha12 = 0.25; gamma1 = 0.2; eta1 = 1
R> alpha21 = 0.3; alpha22 = 0.3; gamma2 = 0.1; eta2 = 1
R> true2 <- list(alpha11 = alpha11, alpha12 = alpha12,
+    gamma1 = gamma1, eta1 = eta1, alpha21 = alpha21,
+    alpha22 = alpha22, gamma2 = gamma2, eta2 = eta2)

#### Setting the sample grid ####
R> n2 <- 50000
R> Time2 <- 1000
R> sam2 <- setSampling(Terminal = Time2, n = n2)

#### Simulation ####
R> yuima2 <- setYuima(model = yuima2, sampling = sam2)
R> set.seed(123)
R> yuima2 <- simulate(yuima2, true.parameter = true2,
+    sampling = sam2)
```

Figure 9 reports the simulated trajectory of each member in the process $X_t := [X_{t,1}, X_{t_2}]^\top$

```
R> plot(yuima2)
```

Now we execute the three-step estimation procedure using, as a dataset, the simulated trajectory stored in the slot `data` of the object `yuima2`. As done in the

univariate case, we select randomly a starting point and we fix upper and lower bounds for each parameter. We select the inputs of the function `qmleLevy` to get an object of `yuima.qmleLevy.incr` class that contains the estimated unit-time increments of the noise and the Lévy measure parameters of the bivariate symmetric Variance Gamma process Z_t.

```
#### Starting point generation ####
R> set.seed(123)
R> start2 <- list(alpha11 = runif(1, 0.01, 2),
+    alpha12 = runif(1, 0.01, 2), gamma1 = runif(1, 0.01, 2),
+    eta1 = runif(1, 0.5, 2), alpha21 = runif(1, 0.01, 2),
+    alpha22 = runif(1, 0.01, 2), gamma2 = runif(1, 0.01, 2),
+    eta2 = runif(1, 0.5, 2))

#### Upper and lower bounds ####
R> upper2 <- list(alpha11 = 2, alpha12 = 2, gamma1 = 2, eta1 = 2,
+    alpha21 = 2, alpha22 = 2, gamma2 = 2, eta2 = 2)
R> lower2 <- list(alpha11 = 0.01, alpha12 = 0.01,
+    gamma1 = 0.01, eta1 = .5, alpha21 = 0.01, alpha22 = 0.01,
+    gamma2 = 0.01, eta2 = .5) ## set lower bound

#### Estimation ####
R> res.VG2 <- qmleLevy(yuima2, start = start2, lower = lower2,
+    upper = upper2, Est.Incr = "IncrPar", aggregation = TRUE,
+    joint = FALSE)
```

With the following command lines, we compare the initial values for the optimization routine, the fixed and estimated parameters.

```
#### Starting values ####
unlist(start2)[names(coef(res.VG2))]

  alpha11   alpha12   alpha21   alpha22    gamma1    gamma2
0.5822793 1.5787272 1.8815299 0.1006574 0.8238641 1.0609299
     eta1      eta2
1.8245261 1.8386286

#### Real parameters ####
unlist(true2)[names(coef(res.VG2))]

alpha11 alpha12 alpha21 alpha22  gamma1  gamma2    eta1    eta2
   0.40    0.25    0.30    0.30    0.20    0.10    1.00    1.00

#### Estimated parameters ####
coef(res.VG2)

  alpha11   alpha12   alpha21   alpha22    gamma1    gamma2
0.3668973 0.2633624 0.2984875 0.3009404 0.2053790 0.1017661
     eta1      eta2
0.9883838 0.9745951
```

The estimated parameters seem to be precise. The Euclidean norm of the difference between `true2` and `coef(res.VG2)` is approximately 0.0457 while, applying the same distance between `true2` and `start2`, it results to be 2.652 with a reduction of 98%. We show the standard errors applying the function `summary`.

```
#### Summary ####
summary(res.VG2)
```

```
summary(res.VG2)
Quasi-Maximum likelihood estimation

Call:
qmleLevy(yuima = yuima2, start = start2, lower = lower2,
    upper = upper2, joint = FALSE, Est.Incr = "IncrPar",
                aggregation = TRUE)

Coefficients:
          Estimate  Std. Error
alpha11  0.3668973 0.005725729
alpha12  0.2633624 0.002610983
alpha21  0.2984875 0.028302434
alpha22  0.3009404 0.017663594
gamma1   0.2053790 0.023555954
gamma2   0.1017661 0.010761713
eta1     0.9883838 0.098752509
eta2     0.9745951 0.094324105

-2 log L: -494221.9 -494557.8 5335.13
```

The estimated time-unit increments of the bivariate Lévy noise are available in the slot `Incr.Lev`

```
R> summary(res.VG2@Incr.Lev)
```

```
   Length1    Length2      Class       Mode
      1000       1000 yuima.data         S4
```

```
R> plot(res.VG2@Incr.Lev,
+    ylab = c(expression(paste(Delta, J[1, t])),
+    expression(paste(Delta, J[2, t]))), xlab = "t")
```

5.3. Real data

In this section, we discuss how to estimate a stochastic differential equation driven by a Lévy process using real data. Once the increments have been obtained, we show how to use them in the two different situations: noise selection and forecasting. We start with an example that shows how to combine

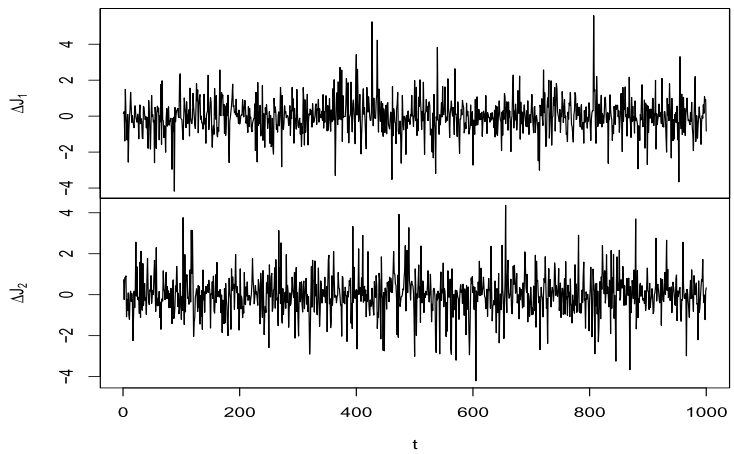

FIG 10. *Estimated unit-time increments of the bivariate process J_t.*

the information stored in an object of `yuima.qmleLevy.incr` class with available R packages for selecting a Lévy measure. The data is downloaded from `yahoo.finance` using the R package `quantmod` that downloads time-series in an `xts` format. We get the closing log-prices of the S&P500 index ranging from 04 January 1951 to 04 January 2021 using the following command lines:

```
#### Download Dataset ####

R> library(quantmod)
R> getSymbols(Symbols = "^GSPC", from = "1951-01-04",
+    to = "2021-01-04")
R> logprice <- log(GSPC$GSPC.Close)
R> plot(logprice, main = "Closed log-prices of Standard & Poor 500",
+    main.cex = 0.8)
```

Figure 11 reports the time series used in our example. We describe the log-price by the following SDE:

$$dX_t = (\alpha_1 + \alpha_2 X_t)\, dt + \gamma_1 X_t^{\gamma_2} dJ_t, \tag{37}$$

where $\alpha_1 > 0$, $\alpha_2 < 0$, $\gamma_1 > 0$ and $\gamma_2 \geq 0$. We construct an object of `yuima` class that contains the mathematical description of the SDE in (37) and the data.

```
#### Law Definition ####
R> mylaw3 <- setLaw(dim = 1)

R> #### Model and Data ####
yuima3 <- setModel(drift = "alpha1+alpha2*X",
+    jump.coeff = matrix(c("gamma1*X^gamma2")),
+    measure.type = "code",
```

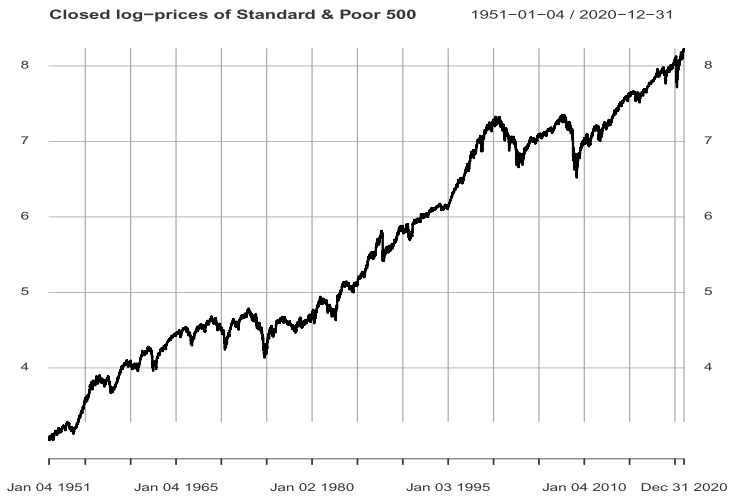

FIG 11. *Closing log-prices of S&P500 ranging from 04 January 1951 to 04 January 2021*

```
+    measure = list(df = mylaw3), jump.variable = "J",
+    solve.variable = c("X"), state.variable = c("X"))
R> Data <- setData(logprice, delta = 1/30)
R> yuima3 <- setYuima(data = Data, model = yuima3)
R> print(Data)

Number of original time series: 1
length = 17615, time range [1951-01-04 ; 2020-12-31]

Number of zoo time series: 1
           length time.min time.max      delta
GSPC.Close  17615        0  587.133 0.03333333
```

From the structure of the object `Data`, we observe that the time is expressed in a monthly basis. Therefore, setting $t_0 = 0$, we have $T_n = 587.133$ and $h = 1/30$.

It is worth noting that the object `mylaw` does not require a formal specification for the random number generator and for the density function as done in the previous examples. Indeed, we do not assume any specific form of the Lévy measure of the process J and the estimation of the unit-time increments described in Section 3 is completely model-free.

```
#### Estimation of time-unit increments ####
R> set.seed(123)
R> start3 <- list(alpha1 = runif(1, min = 10^(-10), max = 1),
+    alpha2 = runif(1, min = -1, max = -10^(-10)),
+    gamma1 = runif(1, min = 10^(-10), max = 1),
+    gamma2 = runif(1, min = 0, max = 2))
R> lower3 <- list(alpha1 = 10^(-10), alpha2 = -1,
```

```
+    gamma1 = 10^(-10), gamma2 =  0)
R> upper3 <- list(alpha1 = 1, alpha2 =  1, gamma1 = 1,
+    gamma2 =  2)
R> res3 <- qmleLevy(yuima3, start = start3, lower = lower3,
+    upper = upper3, Est.Incr = "Incr", aggregation = TRUE,
+    joint = FALSE)
R> summary(res3)
```

```
Quasi-Maximum likelihood estimation

Call:
qmleLevy(yuima = yuima3, start = start3, lower = lower3,
    upper = upper3, joint = FALSE, Est.Incr = "Incr",
    aggregation = TRUE)

Coefficients:
           Estimate   Std. Error
gamma1  0.016267709 0.0007539279
gamma2  0.694168099 0.0715834862
alpha1  0.012156634 0.0081051421
alpha2 -0.000635202 0.0015396235

-2 log L: -113838.7 -113858.5
```

Applying `logLik` method to `res3`, we determine the value for the stepwise GQL function $\mathbb{H}_{1,n}(\hat{\gamma}_1, \hat{\gamma}_2)$ and the value of GQMLE with the following command lines:

```
R> T_n <- tail(index(Data@zoo.data[[1]]),1L)
R> H_1 <- -1/T_n*logLik(res3)[1]
R> GQMLE <-logLik(res3)[2]
R> print(c(H_1, GQMLE))
```

[1] -96.94454 56929.23019

The unit-time increments are stored in the slot `res3@Incr.Lev` and they can be extrapolated using the following command lines:

```
#### Time-unit increments ####
R> UnitaryIncr <- as.numeric(res3@Incr.Lev@original.data)
R> plot(UnitaryIncr, ylab = expression(Delta*J[1]), xlab = " ",
+    main = "Estimated Time-Unit Increments", cex.main = 0.8)
```

Figure 12 reports the estimated time-unit increments of the process J. Due to the fact that the object `UnitaryIncr` belongs to the `numeric` class, we can apply any method available in R for any `numeric` object. Just for an illustration, in the following command lines, we show how to get the kernel density estimate based on the estimated increments, and compare it with the empirical histogram; a graphical comparison is reported in Figure 13.

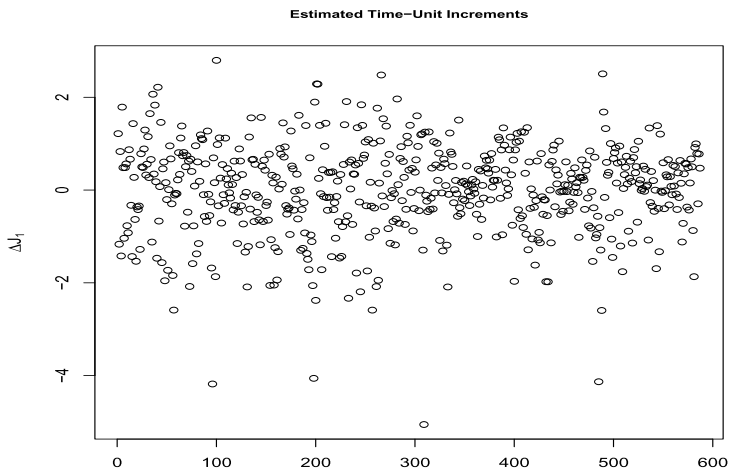

FIG 12. *Estimated time-unit increments from the real dataset*

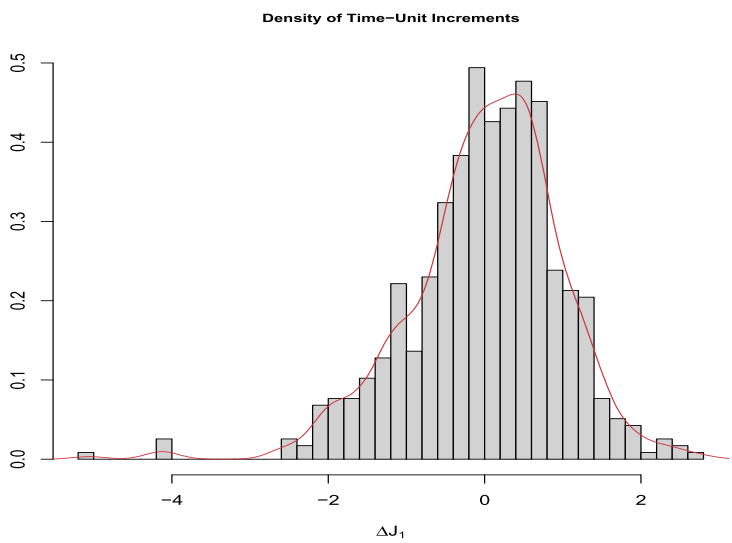

FIG 13. *Comparison betwen empirical and kernel densities of the time-unit increments*

```
#### Plot kernel density ####
R> hist(UnitaryIncr, freq = F, nclass = 50,
+    main = "Density of Time-Unit Increments",
+    cex.main =0.8, xlab = expression(Delta*J[1]), ylab = " ")
R> lines(density(UnitaryIncr),  col = "red")
```

A model selection exercise can be done using the function `stepAIC.ghyp` available in the package `ghyp` that allows the user to compare a list of distribu-

tions widely applied in finance. In particular, based on the Akaike Information Criterion, this function identifies the best model between the Generalized Hyperbolic, the Hyperbolic, the Variance Gamma, the Normal Inverse Gaussian, the Student-t and the Normal distribution.

```
#### Model selection  ####
R> library(ghyp)
R> Comparison <- stepAIC.ghyp(UnitaryIncr)
R> Comparison$best.model
```

```
Asymmetric Hyperbolic Distribution:

Parameters:
 alpha.bar         mu      sigma      gamma
 1.8398103  0.5383745  0.9075450 -0.5337697

log-likelihood:
-793.7689

Call:
stepAIC.ghyp(data = UnitaryIncr)
```

In our example, the function `stepAIC.ghyp` selects the Asymmetric Hyperbolic distribution as the best fitting model with the $(\lambda, \bar{\alpha}, \mu, \sigma, \gamma)$-parametrization for a generic Generalized Hyperbolic distribution. The latter is a normal variance mean mixture with a Generalized Inverse Gaussian subordinator τ_t and, as described in the package documentation [20], the $(\lambda, \bar{\alpha}, \mu, \sigma, \gamma)$-parametrization requires the characteristic function of τ_1 to be:

$$\phi_{\tau_1}(u) = \left(\frac{\varphi}{\varphi - 2iu}\right)^{\frac{\lambda}{2}} \frac{K_\lambda\left(\sqrt{\xi(\varphi - 2iu)}\right)}{K_\lambda\left(\sqrt{\xi\varphi}\right)}$$

where $\varphi = \bar{\alpha}\frac{K_{\lambda+1}(\sqrt{\bar{\alpha}})}{K_\lambda(\sqrt{\bar{\alpha}})}$, $\xi = \bar{\alpha}\frac{K_\lambda(\sqrt{\bar{\alpha}})}{K_{\lambda+1}(\sqrt{\bar{\alpha}})}$ and $\bar{\alpha} > 0$. The real parameters μ and γ control the position and the skewness while $\sigma \geq 0$ is a scale parameter for the Generalized Hyperbolic distribution. The Hyperbolic distribution is obtained by setting $\lambda = 1$.

In the second example, we show how to use an object of `yuima.law` class to generate a new trajectory using the increments of the process J. In this case, we need to estimate the increments associated with the interval of length Δt (small model-time length). In this example, we use a shorter dataset composed of three years of observations of the S&P500 index ranging from 04 January 2018 to 04 January 2021. For the estimation of the increments, the chunk code is exactly the same used in the previous example with only one difference. Indeed, to obtain the Δt increments we set the input `aggregation` as `FALSE`.

```
#### Download dataset ####
R> getSymbols(Symbols = "^GSPC", from = "2018-01-04",
```

```
+    to = "2021-01-04")
R> logprice2 <- log(GSPC$GSPC.Close)

#### Model and data ####
R> mylaw4 <- setLaw(dim = 1)
R> yuima4 <- setModel(drift = "alpha1+alpha2*X",
+    jump.coeff = matrix(c("gamma1*X^gamma2")),
+    measure.type = "code", measure = list(df = mylaw4),
+    jump.variable = "J", solve.variable = c("X"),
+    state.variable = c("X"))
R> Data <- setData(logprice, delta = 1/30)
R> yuima4 <- setYuima(data = Data, model = yuima4)

#### Estimation delta t increments ####
R> set.seed(123)
R> start4 <- list(alpha1 = runif(1, min = 10^(-10), max = 5),
+    alpha2 = runif(1, min = -1, max = -10^(-10)),
+    gamma1 = runif(1, min = 10^(-10), max = 1),
+    gamma2 = runif(1, min = 0, max = 2))
R> lower4 <- list(alpha1 = 10^(-10), alpha2 =  -1,
+    gamma1 = 10^(-10), gamma2 =  0)
R> upper4 <- list(alpha1 = 5, alpha2 =  -10^(-10), gamma1 = 1,
+    gamma2 =  2)
R> res4 <- qmleLevy(yuima4, start = start4, lower = lower4,
+    upper = upper4, Est.Incr = "Incr", aggregation = FALSE,
+    joint = FALSE)
R> summary(res4)

Quasi-Maximum likelihood estimation

Call:
qmleLevy(yuima = yuima4, start = start4, lower = lower4,
    upper = upper4, joint = FALSE, Est.Incr = "Incr",
    aggregation = FALSE)

Coefficients:
          Estimate  Std. Error
gamma1  0.08111166 0.005677789
gamma2  0.00000000 0.065495083
alpha1  2.01137182 1.367347271
alpha2 -0.25019063 0.171158938

-2 log L: -4207.318 -4210.077
```

Using the estimated Δt increments in the slot `res4@Incr.Lev` we can build an object of `yuima.law` class that internally uses a random number generator that samples from the data in `res4@Incr.Lev`.

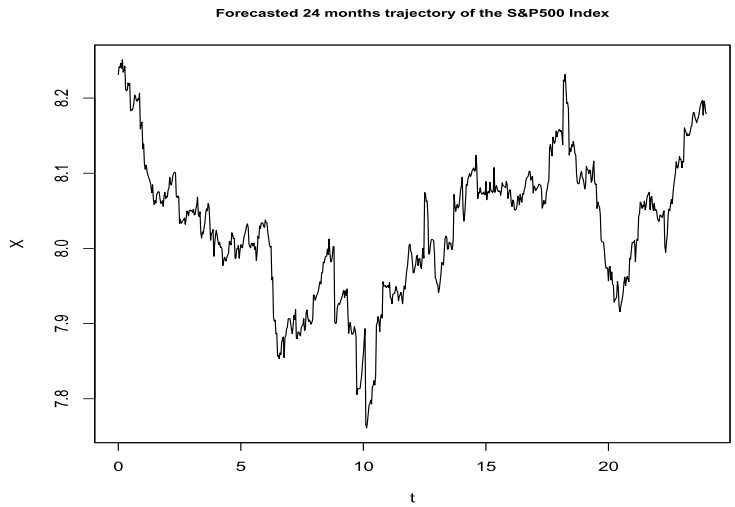

FIG 14. *24 months simulated trajectory of the S&P500 log-price series*

```
#### yuima.law Definition ####
R> mydata <- as.numeric(res4@Incr.Lev@original.data)
R> myrndEmp <- function(n, mydata){
+    sample(mydata, size = n)
+  }
R> mylaw5 <- setLaw(rng = myrndEmp)
```

The object `mylaw` contains a random number generator that uses the R function `sample`, however, the user can apply more advanced sampling methods from packages available from CRAN. We can simulate one-year trajectory of the S&P500 log prices in YUIMA. Figure 14 reports the simulated sample path.

```
#### Generation 1 year trajectory ####
R> yuima5 <- setModel(drift = "alpha1+alpha2*X",
+    jump.coeff = matrix(c("gamma1*X^gamma2")),
+    measure.type = "code", measure = list(df = mylaw5),
+    jump.variable = "J", solve.variable = c("X"),
+    state.variable = c("X"),
+    xinit = as.numeric(tail(logprice2, 1L)))
R> samp5 <- setSampling(Initial = 0, Terminal = 24, n = 24*30)
R> yuima5 <- setYuima(model = yuima5, sampling = samp5)
R> true5 <- as.list(coef(res4))
R> true5$mydata <- mydata
R> set.seed(123)
R> yuima5 <- simulate(yuima5, true.parameter = true5,
+    sampling = samp5)
R> plot(yuima5,
```

```
+    main = "Forecasted 24 months trajectory of the S&P500 Index",
+    cex.main = 0.8)
```

In Figure 14, the 24 months trajectory displays an oscillatory behavior. It fluctuates around the long term mean that can be estimated using the ratio $-\frac{\hat{\alpha_1}}{\hat{\alpha_2}} \approx 8.0394$.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work was partly supported by JST CREST Grant Number JPMJCR14D7, Japan.

References

[1] R. A. Adams. Some integral inequalities with applications to the imbedding of Sobolev spaces defined over irregular domains. *Trans. Amer. Math. Soc.*, 178:401–429, 1973. MR0322494

[2] A. Brouste, M. Fukasawa, H. Hino, S. M. Iacus, K. Kamatani, Y. Koike, H. Masuda, R. Nomura, T. Ogihara, Y. Shimizu, M. Uchida, and N. Yoshida. The yuima project: A computational framework for simulation and inference of stochastic differential equations. *Journal of Statistical Software*, 57(4):1–51, 2014.

[3] V. Genon-Catalot and J. Jacod. On the estimation of the diffusion coefficient for multi-dimensional diffusion processes. *Ann. Inst. H. Poincaré Probab. Statist.*, 29(1):119–151, 1993. MR1204521

[4] S. M. Iacus and L. Mercuri. Implementation of lévy carma model in yuima package. *Comp. Stat.*, 30(4):1111–1141, 2015. MR3433445

[5] S. M. Iacus, L. Mercuri, and E. Rroji. Cogarch (p, q): Simulation and inference with the yuima package. *J. Stat. Softw.*, 80(1):1–49, 2017.

[6] S. M. Iacus, L. Mercuri, and E. Rroji. Discrete-time approximation of a cogarch (p, q) model and its estimation. *J. Time Ser. Anal.*, 39(5):787–809, 2018. MR3849527

[7] S. M. Iacus and N. Yoshida. Simulation and inference for stochastic processes with yuima. *A comprehensive R framework for SDEs and other stochastic processes. Use R*, 2018. MR3793179

[8] M. Kessler. Estimation of an ergodic diffusion from discrete observations. *Scandinavian Journal of Statistics* 24(2): 211–229, 1997. MR1455868

[9] U. Küchler and S. Tappe. Bilateral gamma distributions and processes in financial mathematics. *Stochastic Process. Appl.*, 118(2):261–283, 2008. MR2376902

[10] A. Løkka. Martingale representation of functionals of Lévy processes. *Stochastic Process. Appl.*, 22(4): 867–892, 2004. MR2062949

[11] D. B. Madan and E. Seneta. The variance gamma (v.g.) model for share market returns. *J. Bus.*, 63(4):511–524, 1990.

[12] H. Masuda. Convergence of Gaussian quasi-likelihood random fields for ergodic Lévy driven SDE observed at high frequency. *Ann. Statist.*, 41(3):1593–1641, 2013. MR3113823

[13] H. Masuda and Y. Uehara. Two-step estimation of ergodic Lévy driven SDE. *Stat. Inference Stoch. Process.*, 20(1):105–137, 2017. MR3619574

[14] P. E. Protter, *Stochastic Integration and Differential Equations, second edition*, Springer-Verlag, Berlin. MR2020294

[15] K.-i. Sato. *Lévy processes and infinitely divisible distributions*. Cambridge university press, 1999. MR1739520

[16] D. Scott and C. Y. Dong. *VarianceGamma: The Variance Gamma Distribution*, 2018. R package version 0.4-0.

[17] D. Straumann. *Estimation in conditionally heteroscedastic time series models*, volume 181 of *Lecture Notes in Statistics*. Springer-Verlag, Berlin, 2005. MR2142271

[18] Y. Uehara and H. Masuda. Stepwise estimation of a Lévy driven stochastic differential equation. *Proc. Inst. Statist. Math. (Japanese)*, 65(1):21–38, 2017. MR3701830

[19] Y. Uehara, Statistical inference for misspecified ergodic Lévy driven stochastic differential equation models. *Stochastic Process. Appl.*, 129(10): 4051–4081, 2019. MR3997671

[20] M. Weibel, D. Luethi, and W. Breymann. *ghyp: Generalized Hyperbolic Distribution and Its Special Cases*, 2020. R package version 1.6.1.