Neural Fractional Stochastic Differential Equations

Yongxin Yang
King's College London
yongxin.yang@kcl.ac.uk

Timothy Hospedales
University of Edinburgh
t.hospedales@ed.ac.uk

Abstract

Fractional differential equations (FDEs) model systems with long-range memory, while stochastic differential equations (SDEs) capture inherent randomness. Existing neural differential equation models address either memory (Neural FDEs) or stochasticity (Neural SDEs), but a unified framework combining both with a learnable fractional order and an efficient adjoint method for backpropagation remains absent. We introduce the Neural Fractional Stochastic Differential Equation (Neural FSDE), which learns continuous-time dynamics with both memory and randomness from data via a Caputo FSDE with neural drift, neural diffusion, and a learnable fractional order. To enable scalable training, we derive a discrete adjoint method that computes gradients with respect to the initial state, the drift and diffusion network parameters, and the fractional order, with memory cost linear in the number of time steps, against quadratic memory for backpropagation through the solver via automatic differentiation. We validate the framework on two real-world problems. On the COVID-19 model of Bonyah et al. [10], it gives the first end-to-end calibration on real data and recovers a fractional order below 1. On S&P 500 index options, the Neural FSDE attains the lowest implied-volatility error among rough Bergomi, rough Heston, path-dependent volatility, and Neural SDE models. In particular, the Neural FSDE outperforms the Neural SDE with the same drift and diffusion functions on every instrument and evaluation schedule, and this demonstrates the contribution of the fractional order. Code is available at `https://github.com/qmfin/NeuralFSDE`.

1 Introduction

Ordinary differential equations (ODEs) describe deterministic temporal dynamics, partial differential equations (PDEs) extend these to spatiotemporal phenomena, and stochastic differential equations (SDEs) introduce randomness for systems with inherent uncertainty or noise. Fractional differential equations (FDEs) [19] replace integer-order time derivatives with non-integer ones, capturing systems with memory and non-locality where the future state depends on the entire trajectory of past states. Examples include viscoelastic materials, anomalous diffusion, and long-range dependencies in financial time series.

The intersection of deep learning and differential equations began with the Neural ODE [11], which replaces the discrete layers of a neural network with a continuous-time dynamical system whose vector field is itself a neural network. The paradigm has since been extended to neural PDEs [58], Neural SDEs [42, 40], and Neural FDEs [12, 57, 38]. A model that combines the SDE and FDE branches, and so exhibits both memory and randomness, has not been studied end-to-end with an efficient backpropagation method.

In this work, we introduce the Neural Fractional Stochastic Differential Equation (Neural FSDE), parametrising the drift and diffusion of a Caputo FSDE by neural networks alongside a learnable fractional order. To make training practical with long time grids and complex neural drift and diffusion networks, we derive a discrete adjoint method whose memory cost grows linearly with the number of time steps and which provides gradients for the initial state, the network parameters, and the

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

fractional order. We validate the framework on two real-world problems. On the stochastic-fractional COVID-19 model of Bonyah et al. [10], we provide the first end-to-end real-data calibration of that FSDE. In S&P 500 index option pricing, we recast the rough-volatility models rough Bergomi and rough Heston, a Neural SDE, and the Neural FSDE as Caputo-FSDE special cases, calibrate them in a single pipeline and compare them with a path-dependent volatility model [30]. The Neural FSDE attains the lowest median implied-volatility RMSE (IV-RMSE) in all eight settings formed by two instruments (SPX weekly and SPY options) and four evaluation schedules (Section 4.3).

2 Related Work

Neural ODEs [11] introduced continuous-depth models trained via the adjoint sensitivity method. Subsequent refinements cover dimension augmentation to recover topology lost under integer-order flows [21], reverse-time-accurate adjoints that mitigate the forward/backward trajectory divergence of the original continuous adjoint [56, 45], effectively symmetric Runge-Kutta schemes for stable reversible solvers [51], latent ODEs and ODE-RNN for irregularly sampled time series [50], controlled differential equations driven by interpolated paths [39], and physics-informed neural-PDE solvers [58]. Two branches have since followed: (1) Neural SDEs [42, 40] attach a Wiener-driven diffusion for stochastic dynamics, with later work on adversarial training as infinite-dimensional GANs [41], distributional stability under irregular observations [47], and the score-based reverse-time SDE paradigm in generative modelling [52]. (2) Neural FDEs [12, 57, 38] replace the integer-order time derivative with a fractional one to capture long-range memory; the Caputo design has been extended to graph neural networks under the FROND framework [37] and to learnable variable-order $\alpha(t, x)$ [13]. The Neural FSDE combines these two branches, attaching a Wiener diffusion to a Caputo-fractional state derivative; this combination has not been studied end-to-end with an efficient adjoint method for backpropagation.

Two prior lines of work are most closely related but differ structurally. fSDE-Net [32] replaces the Brownian motion of a Neural SDE with a fractional Brownian motion (fBM): $\mathrm{d}X_t = b_\theta(t, X_t)\,\mathrm{d}t + \sigma_\theta(t, X_t)\,\mathrm{d}B_t^H$; related work parametrises the fractional white-noise increment with a sparse-Gaussian-process approximation [53], extends the design to fractional generative diffusion models [46], or learns the Hurst index under a Markov approximation of fBM with a time-dependent H [14]. Despite the names, in all of these the time derivative remains integer-order and only the noise carries memory, so the model is mathematically an ODE perturbed by a non-Markovian noise process, and backpropagation reduces to the standard Neural ODE adjoint. The Neural FSDE, by contrast, is a Caputo FSDE in which the singular kernel $(t-s)^{\alpha-1}$ acts on both drift and diffusion. Its adjoint (Section 3.3) therefore needs two accumulators, weighted sums of future gradients whose weights scale as $(\Delta t)^\alpha$ for the drift kernel and as $(\Delta t)^{\alpha-1/2}$ for the diffusion kernel. It also needs an additional gradient with respect to the fractional order α. Neural SVEs [8] parametrise the convolutional kernels of a stochastic Volterra equation with neural networks, generalising the affine Volterra family [2] that underlies rough-Heston and rough-Bergomi tractability. Our Caputo Neural FSDE is a strict special case of the Neural SVE class, obtained by fixing $K_\mu(\tau) = K_\sigma(\tau) = \tau^{\alpha-1}/\Gamma(\alpha)$ and learning a single scalar memory exponent α in place of two general kernels, so the SVE class is more expressive in principle. Neural SVEs are, however, trained by automatic differentiation through the discretised solver, the continuous adjoint being infeasible for their non-Markovian structure [8, Remark 3.2], and their experiments use $N{=}50$ time steps. For the Caputo case we derive a discrete adjoint whose memory grows linearly with the number of time steps (Section 3.3), which makes long time grids with complex neural drift and diffusion networks practical (Section 4.1). The Caputo kernel is itself a mixture of exponentially decaying modes, as in state-space models such as S4 [29] and Mamba [28], but all its weights are set by the single exponent α (Appendix B).

Fractional dynamics arise in applications with long-range memory. In finance, log-volatility behaves empirically as a fractional Brownian motion with Hurst index $H \approx 0.1$ [24], motivating rough-volatility models [5, 22], whose deep-learning calibration has either inverted a neural surrogate of the model-to-implied-volatility map [6, 34, 23, 4] or fitted a Markovian SDE end-to-end through a differentiable simulator [25]; we extend the latter line to fractional dynamics. In epidemiology, fractional-order compartmental models [16, 10] and fractional-Brownian SIR fits [20] have been applied to COVID-19, with memory representing the lingering influence of past events such as incubation periods and public-health interventions.

3 Method

3.1 Caputo-type Fractional Stochastic Differential Equation

We begin with the classical left-sided Riemann-Liouville fractional integral of order α for a function $X(t)$ over the interval $(0, T)$ [44]:

$$I^{\alpha}X(t) := \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{\alpha-1} X(s)\mathrm{d}s, \quad t \in (0, T) \tag{1}$$

Here, α is a positive real number. For the case where $0 < \alpha \leq 1$, the Riemann-Liouville fractional derivative is constructed by first applying a fractional integral of order $1-\alpha$ and then a standard first-order derivative:

$${}^{\mathrm{RL}}_{0}D^{\alpha}_{t}X(t) := \frac{\mathrm{d}}{\mathrm{d}t}\left(I^{1-\alpha}X(t)\right) = \frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha}X(s)\mathrm{d}s \tag{2}$$

An alternative definition, the Caputo fractional derivative, reverses the order of these operations: it first takes a standard first-order derivative and subsequently applies a fractional integral of order $1-\alpha$:

$${}^{\mathrm{C}}_{0}D^{\alpha}_{t}X(t) := I^{1-\alpha}\left(X'(t)\right) = \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha}X'(s)\mathrm{d}s \tag{3}$$

We adopt the Caputo-type derivative due to two desirable properties: (1) the derivative of a constant is zero, and (2) the initial condition is defined via integer-order values, such as $X(0)$.

A Caputo-type fractional stochastic differential equation (FSDE) is written formally as

$${}^{\mathrm{C}}_{0}D^{\alpha}_{t}X(t) = \mu(t, X(t)) + \sigma(t, X(t))\frac{\mathrm{d}W(t)}{\mathrm{d}t}, \quad X(0) = X_0 \tag{4}$$

in terms of a drift function $\mu(t, X(t))$ and a diffusion function $\sigma(t, X(t))$. We assume that $\mu : [0, T] \times \mathbb{R}^D \to \mathbb{R}^D$ and $\sigma : [0, T] \times \mathbb{R}^D \to \mathbb{R}^D$ (i.e., diagonal noise), and that $W(t)$ is a D-dimensional Wiener process with independent components. Eq. 4 is a formal expression: a Caputo derivative requires $X'(t)$, which does not exist for an SDE solution (only Hölder-continuous a.s.). The rigorous definition of the process is the integral form

$$X(t) = X(0) + \frac{1}{\Gamma(\alpha)}\int_0^t (t-s)^{\alpha-1}\mu(s, X(s))\mathrm{d}s + \frac{1}{\Gamma(\alpha)}\int_0^t (t-s)^{\alpha-1}\sigma(s, X(s))\mathrm{d}W(s) \tag{5}$$

where the last term is an Itô integral. Square integrability of the kernel $(t-s)^{\alpha-1}$ requires $\alpha > 1/2$, so we restrict α to the range $(1/2, 1]$ throughout this paper. At $\alpha = 1$ the kernel is constant and Eq. 5 is an Itô SDE, the memoryless case.

3.2 Euler-Maruyama Scheme

To numerically solve Eq. 5, we adopt the Euler-Maruyama scheme [19]. We first partition the interval $[0, T]$ into N equal subintervals of width $\Delta t = T/N$, with time steps $\tau_n = n\Delta t$ for $n \in \{0, 1, \ldots, N\}$. At a specific time τ_{n+1}, the solution is given by:

$$X(\tau_{n+1}) = X(0) + \frac{1}{\Gamma(\alpha)}\sum_{j=0}^{n}\int_{\tau_j}^{\tau_{j+1}} (\tau_{n+1}-s)^{\alpha-1}\mu(s, X(s))\mathrm{d}s \tag{6}$$

$$+ \frac{1}{\Gamma(\alpha)}\sum_{j=0}^{n}\int_{\tau_j}^{\tau_{j+1}} (\tau_{n+1}-s)^{\alpha-1}\sigma(s, X(s))\mathrm{d}W(s) \tag{7}$$

On each subinterval $[\tau_j, \tau_{j+1}]$ we apply a left-point approximation: in the drift term, $\mu(s, X(s))$ is replaced by $\mu(\tau_j, Y_j)$, where Y_j approximates $X(\tau_j)$, and the kernel is integrated exactly; in the diffusion term, both $\sigma(s, X(s))$ and the singular kernel $(\tau_{n+1}-s)^{\alpha-1}$ are evaluated at $s = \tau_j$. The left-point diffusion rule introduces a variance error in the $j = n$ term whose constant is $\mathcal{O}(1/(2\alpha-1))$

and worsens as α approaches $\frac{1}{2}$, consistent with the convergence theory in [19]. These approximations yield the following discretised scheme for the trajectory $[Y_0, Y_1, \ldots, Y_N]$:

$$Y_{n+1} = Y_0 + \frac{(\Delta t)^\alpha}{\Gamma(\alpha+1)} \sum_{j=0}^{n} \left((n+1-j)^\alpha - (n-j)^\alpha\right) \mu(\tau_j, Y_j) + \frac{(\Delta t)^{\alpha-1/2}}{\Gamma(\alpha)} \sum_{j=0}^{n} (n+1-j)^{\alpha-1} \left(\sigma(\tau_j, Y_j) \odot Z_j\right) \tag{8}$$

with $Y_0 = X(0)$, independent $Z_j \sim \mathcal{N}(0, I)$ such that $\sqrt{\Delta t}\, Z_j$ is the Brownian increment over $[\tau_j, \tau_{j+1}]$, and $\odot$ the elementwise product. The scheme is non-local, that is, it has *memory*: computing Y_{n+1} requires the entire history of the trajectory up to time τ_n. In each iteration, the drift and diffusion functions are evaluated only once at the current time step, and previous evaluations are reused, weighted differently for each new step. The non-locality gives the solver $\mathcal{O}(N^2)$ time per trajectory; the standard autograd backward pass also requires $\mathcal{O}(N^2)$ memory per path and state dimension through the pairwise dependency graph. This motivates the discrete adjoint of Section 3.3, which reduces the memory to $\mathcal{O}(N \cdot D \cdot B)$ for B simulated paths of the D-dimensional state. Eq. 8 can be viewed as a weighted sum of historical drift and diffusion terms. For notational simplicity, we define the weights

$$w_i^\mu = \frac{(\Delta t)^\alpha}{\Gamma(\alpha+1)}((i+1)^\alpha - i^\alpha), \qquad w_i^\sigma = \frac{(\Delta t)^{\alpha-1/2}}{\Gamma(\alpha)}(i+1)^{\alpha-1} \tag{9}$$

Eq. 8 can then be rewritten more compactly as:

$$Y_{n+1} = Y_0 + \sum_{j=0}^{n} w_{n-j}^\mu \mu(\tau_j, Y_j) + \sum_{j=0}^{n} w_{n-j}^\sigma \left(\sigma(\tau_j, Y_j) \odot Z_j\right) \tag{10}$$

For the deterministic Caputo counterpart, Adams-type predictor-corrector schemes [18] are the standard alternative; complexity-reducing approximations include sum-of-exponentials decompositions of the singular kernel $t^{\alpha-1}$ that reduce cost to nearly $\mathcal{O}(N \log N)$ [36, 55], and hybrid schemes used in rough-volatility simulation [7]. The discrete adjoint we derive next is specific to the Euler-Maruyama discretisation in Eq. 8. In Appendix B we check at synthetic scale that the recovered fractional order is unchanged under a predictor-corrector forward, and implement a sum-of-exponentials forward whose mode weights are explicit differentiable functions of α, lowering the forward cost to $\mathcal{O}(NK)$ for K modes; an adjoint for either scheme is left for future work.

3.3 Neural FSDE and Adjoint Method

The drift and diffusion functions in Eq. 4 can be parametrised by neural networks with parameters θ and ϕ, respectively, leading to a Neural FSDE:

$${}_0^{\mathrm{C}}D_t^\alpha X(t) = \mu_\theta(t, X(t)) + \sigma_\phi(t, X(t)) \frac{\mathrm{d}W(t)}{\mathrm{d}t}, \quad X(0) = X_0, \quad \alpha \in (1/2, 1] \tag{11}$$

We also treat α as a learnable scalar. To respect the well-posedness range $\alpha \in (1/2, 1]$ established in Section 3.1, we parametrise it as $\alpha = \frac{1}{2} + \frac{1}{2}\,\mathrm{sigmoid}(u_\alpha)$, where $u_\alpha \in \mathbb{R}$ is the unconstrained variable; the loss gradient with respect to u_α then follows by the chain rule. The forward pass of a Neural FSDE is computed using a numerical solver such as the Euler-Maruyama scheme described above. While this solver is differentiable, backpropagating through the entire computation graph needs memory quadratic in the number of steps and is slow on long grids (Section 4.1). For deterministic fractional PDEs, automatic differentiation cannot evaluate the Caputo derivative of a network output, so fractional physics-informed neural networks [48] discretise it with the L1 finite-difference scheme and minimise a PDE residual rather than fitting sample paths; for the SDE we instead derive a discrete adjoint method for memory-efficient gradient computation.

A typical adjoint method avoids storing the entire forward trajectory by reconstructing states in reverse. However, this is challenging for Neural FSDEs due to their non-local, history-dependent nature. Consequently, we store the full state trajectory $[Y_0, Y_1, \ldots, Y_N]$ and the noise samples $[Z_0, Z_1, \ldots, Z_{N-1}]$ during the forward pass. Techniques such as the virtual Brownian tree [42] could avoid storing the noise path, but we do not explore them here.

Without loss of generality, we consider a loss function $L(Y_0, Y_1, \ldots, Y_N)$ that may depend on any of the states in the trajectory. Our goal is to compute the gradients of this loss with respect to the initial state, the network parameters θ and ϕ, and the fractional order α.

Throughout this paper, gradients of a scalar with respect to a vector are row vectors. A Jacobian $\partial f/\partial x$ then has rows indexed by the output and columns by the input, and vector-matrix products read left to right. We define the *total gradient*, or adjoint, $a(t) = \frac{\mathrm{d}L}{\mathrm{d}X(t)}$ as the total sensitivity of the final loss to the state at time t. This quantity captures both the direct contribution of the state to the loss and its indirect influence on the future trajectory via the memory kernel. In the discrete setting we write $a_n = \frac{\mathrm{d}L}{\mathrm{d}Y_n}$, where d denotes this total derivative and ∂ the partial derivative with all other states held fixed. Applying the chain rule, we establish a backward recurrence for the total gradient:

$$a_n = \frac{\mathrm{d}L}{\mathrm{d}Y_n} = \frac{\partial L}{\partial Y_n} + \sum_{j=n+1}^{N} \frac{\mathrm{d}L}{\mathrm{d}Y_j}\frac{\partial Y_j}{\partial Y_n} = \frac{\partial L}{\partial Y_n} + \sum_{j=n+1}^{N} a_j \frac{\partial Y_j}{\partial Y_n} \tag{12}$$

To evaluate the partial derivative $\frac{\partial Y_j}{\partial Y_n}$ for $1 \le n < j$, we differentiate Eq. 10, yielding:

$$\frac{\partial Y_j}{\partial Y_n} = w^{\mu}_{j-1-n}\frac{\partial \mu(\tau_n, Y_n)}{\partial Y_n} + w^{\sigma}_{j-1-n}\,\mathrm{Diag}(Z_n)\frac{\partial \sigma(\tau_n, Y_n)}{\partial Y_n} \tag{13}$$

Substituting Eq. 13 into Eq. 12, we observe that the backward recurrence involves weighted sums of future gradients. To streamline the notation, we introduce two accumulation variables, λ^{μ}_n and λ^{σ}_n:

$$\lambda^{\mu}_n = \sum_{j=n+1}^{N} w^{\mu}_{j-1-n} a_j, \qquad \lambda^{\sigma}_n = \sum_{j=n+1}^{N} w^{\sigma}_{j-1-n} a_j \tag{14}$$

The two accumulators play distinct roles. The accumulator λ^{μ}_n discretises a deterministic right-sided fractional integral: as $\Delta t \to 0$ with $\tau_n \to t$, $\lambda^{\mu}_n \to \frac{1}{\Gamma(\alpha)}\int_t^T (s-t)^{\alpha-1} a(s)\,\mathrm{d}s = {}_tI^{\alpha}_T[a(t)]$, the right-sided Riemann-Liouville integral.

In contrast, λ^{σ}_n enters the gradient only through the product $\lambda^{\sigma}_n \odot Z_n$; this product is the discrete counterpart of the stochastic adjoint contribution, with the kernel scaling and the Brownian increment scaling absorbed into a single weight w^{σ}. Besides differing in discrete scaling ($w^{\mu}_i \propto (\Delta t)^{\alpha}$ vs. $w^{\sigma}_i \propto (\Delta t)^{\alpha-1/2}$), the two weight sequences are not proportional, because the Euler-Maruyama scheme treats the drift kernel exactly but the diffusion kernel via a left-point rule; a single λ is therefore insufficient. Using these definitions, and starting from $a_N = \frac{\partial L}{\partial Y_N}$, the total gradient a_n for $1 \le n \le N-1$ can be computed recursively backward in time using

$$a_n = \frac{\partial L}{\partial Y_n} + \lambda^{\mu}_n \frac{\partial \mu(\tau_n, Y_n)}{\partial Y_n} + (\lambda^{\sigma}_n \odot Z_n)\,\frac{\partial \sigma(\tau_n, Y_n)}{\partial Y_n} \tag{15}$$

The adjoint at the initial time, a_0, requires special consideration because the initial state Y_0 directly influences every subsequent state in the trajectory:

$$\begin{aligned} a_0 &= \frac{\partial L}{\partial Y_0} + \sum_{j=1}^{N} a_j \frac{\partial Y_j}{\partial Y_0} \\ &= \frac{\partial L}{\partial Y_0} + \sum_{j=1}^{N} a_j \left(I + w^{\mu}_{j-1}\frac{\partial \mu(\tau_0, Y_0)}{\partial Y_0} + w^{\sigma}_{j-1}\,\mathrm{Diag}(Z_0)\frac{\partial \sigma(\tau_0, Y_0)}{\partial Y_0} \right) \\ &= \frac{\partial L}{\partial Y_0} + \lambda^{\mu}_0 \frac{\partial \mu(\tau_0, Y_0)}{\partial Y_0} + (\lambda^{\sigma}_0 \odot Z_0)\,\frac{\partial \sigma(\tau_0, Y_0)}{\partial Y_0} + \sum_{j=1}^{N} a_j \end{aligned} \tag{16}$$

The resulting expression for a_0 is, by definition, the gradient of the loss with respect to the initial input $X(0)$.

Having computed the adjoint states, we can derive the gradients for the network parameters. The gradient for θ is the sum of contributions from each time step. For a single step n, substituting λ^{μ}_n yields:

$$g^{\theta}_n = \frac{\mathrm{d}L}{\mathrm{d}\mu(\tau_n, Y_n)}\frac{\partial \mu(\tau_n, Y_n)}{\partial \theta} = \left(\sum_{j=n+1}^{N} \frac{\mathrm{d}L}{\mathrm{d}Y_j}\frac{\partial Y_j}{\partial \mu(\tau_n, Y_n)} \right) \frac{\partial \mu(\tau_n, Y_n)}{\partial \theta} = \lambda^{\mu}_n \frac{\partial \mu(\tau_n, Y_n)}{\partial \theta} \tag{17}$$

Similarly, the contribution to the gradient for ϕ at step n can be expressed:

$$g_n^\phi = \frac{\mathrm{d}L}{\mathrm{d}\sigma(\tau_n, Y_n)} \frac{\partial \sigma(\tau_n, Y_n)}{\partial \phi} = \left(\sum_{j=n+1}^{N} \frac{\mathrm{d}L}{\mathrm{d}Y_j} \frac{\partial Y_j}{\partial \sigma(\tau_n, Y_n)} \right) \frac{\partial \sigma(\tau_n, Y_n)}{\partial \phi} = (\lambda_n^\sigma \odot Z_n) \frac{\partial \sigma(\tau_n, Y_n)}{\partial \phi} \tag{18}$$

Finally, we compute the gradient of the loss with respect to the fractional order α. This involves differentiating the weights w^μ and w^σ with respect to α and accumulating the contributions from each step n. Treating $a_j \in \mathbb{R}^D$ as a vector and $\langle \cdot, \cdot \rangle$ as the Euclidean inner product:

$$\begin{aligned} g_n^\alpha &= \sum_{j=n+1}^{N} \left\langle a_j,\, \frac{\partial Y_j}{\partial w_{j-1-n}^\mu} \frac{\mathrm{d}w_{j-1-n}^\mu}{\mathrm{d}\alpha} + \frac{\partial Y_j}{\partial w_{j-1-n}^\sigma} \frac{\mathrm{d}w_{j-1-n}^\sigma}{\mathrm{d}\alpha} \right\rangle \\ &= \sum_{j=n+1}^{N} \langle a_j,\, \mu(\tau_n, Y_n) \rangle \frac{\mathrm{d}w_{j-1-n}^\mu}{\mathrm{d}\alpha} + \langle a_j,\, \sigma(\tau_n, Y_n) \odot Z_n \rangle \frac{\mathrm{d}w_{j-1-n}^\sigma}{\mathrm{d}\alpha} \\ &= \left\langle \sum_{j=n+1}^{N} \frac{\mathrm{d}w_{j-1-n}^\mu}{\mathrm{d}\alpha} a_j,\, \mu(\tau_n, Y_n) \right\rangle + \left\langle \sum_{j=n+1}^{N} \frac{\mathrm{d}w_{j-1-n}^\sigma}{\mathrm{d}\alpha} a_j,\, \sigma(\tau_n, Y_n) \odot Z_n \right\rangle \end{aligned} \tag{19}$$

The derivatives of the weights involve the digamma function, $\psi(z) = \frac{\Gamma'(z)}{\Gamma(z)}$. The derivative of the drift weight w_i^μ is, with the convention $0^\alpha \ln(0) = 0$ for $i = 0$:

$$\frac{\mathrm{d}w_i^\mu}{\mathrm{d}\alpha} = (\ln(\Delta t) - \psi(\alpha + 1))\, w_i^\mu + \frac{(\Delta t)^\alpha}{\Gamma(\alpha + 1)} \left((i+1)^\alpha \ln(i+1) - i^\alpha \ln(i) \right) \tag{20}$$

and the derivative of the diffusion weight w_i^σ is:

$$\frac{\mathrm{d}w_i^\sigma}{\mathrm{d}\alpha} = (\ln(\Delta t) + \ln(i+1) - \psi(\alpha))\, w_i^\sigma \tag{21}$$

In summary, the gradients for θ, ϕ, and α are computed via a single reverse-time loop from $n = N$ down to 0. In each iteration n, we update the total gradient a_n (Eq. 15) and calculate the local contributions g_n^θ, g_n^ϕ, and g_n^α, accumulating them into their respective parameter gradients.

4 Experiments

We evaluate the Neural FSDE first on synthetic data to verify the discrete adjoint and the fractional-order gradient, then on two real-world problems: COVID-19 epidemic modelling under Bonyah's stochastic-fractional compartmental model, and S&P 500 index option pricing under stochastic volatility.

4.1 Synthetic Data

Toy α-recovery. We test whether the Neural FSDE can simultaneously recover the drift, diffusion, and fractional order of a known FSDE from sample paths. The truth is $\mu(X) = \cos(X)$, $\sigma(X) = \sin(X)$, $\alpha = 0.75$, $X(0) = 0$, discretised with $N = 100$ steps over $[0, 1]$ and $B = 1024$ truth paths. The model uses time-invariant 3-layer MLP heads $\mu_\theta(x), \sigma_\phi(x)$ (hidden 64, $\tanh$). Being time-invariant like the truth, the networks cannot compensate for an incorrect α through spurious time dependence. We initialise α at 0.55, near the lower well-posedness boundary $\frac{1}{2}$, under the parametrisation of Section 3.3. We minimise the 1-Wasserstein distance between generated and truth samples at each time step, averaged over steps (closed-form for 1-D distributions: mean absolute difference between sorted samples), for 2000 Adam steps, with separate learning rates 5×10^{-3} for the network parameters and 2×10^{-2} for u_α. The path mean envelope and marginal densities at $t \in \{0.10, 0.25, 0.50, 0.99\}$ match the truth closely (Figure 1, left and middle), and $\hat{\alpha}$ stabilises tightly around the truth, at a final $\hat{\alpha} = 0.7516$ (target 0.75, 0.2% recovery error), so the adjoint gradient for α provides a usable identification signal.

Adjoint vs. autograd. We benchmark our discrete adjoint against direct PyTorch backpropagation through the same Euler-Maruyama forward, on a single NVIDIA H100 GPU with 3-layer MLP drift

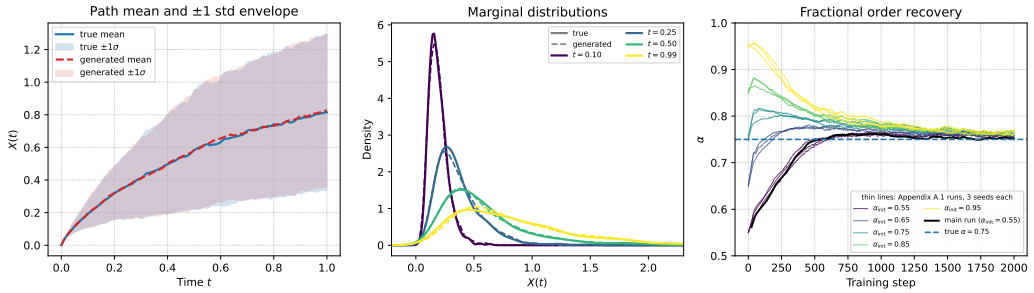


Figure 1: Toy α-recovery. Left: path mean and $\pm 1\sigma$ envelope, true vs. generated. Middle: marginal densities at four times. Right: α over training for the main run (black, initialised at 0.55) and the 15 runs of Appendix A.1 (five initialisations, three seeds each); dashed: the true value 0.75.

and diffusion. At $N{=}1024$, the adjoint uses $44\times$ less peak memory (≈ 25 MB vs. $\approx 1{,}120$ MB) and runs $\approx 5\times$ faster, since traversing the autograd graph at that scale dominates the cost. At $N{=}128$ the memory advantage also holds when the number of paths B or the hidden width H grows instead, at $14\times$ for $B{=}1024$ and $4\times$ for $H{=}512$. The adjoint gradients for X_0, θ, ϕ, and α agree with autograd to within $\sim 10^{-6}$ at five values of α in $[0.63, 0.98]$. Time-and-memory sweeps versus N, B, and H are in Appendix A (Figures 2–4). The adjoint leaves the forward cost unchanged: a Neural SDE step costs $\mathcal{O}(1)$, whereas each Caputo step convolves the whole stored history, so the forward pass remains $\mathcal{O}(N^2)$ per path and the adjoint's benefit lies entirely in the backward pass, where it removes the $\mathcal{O}(N^2)$ autograd graph and its traversal (Appendix B summarises the costs).

4.2 COVID-19 Epidemic With Memory and Randomness

Setup. We apply our framework to the five-compartment Caputo FSDE COVID-19 model of Bonyah et al. [10] on $X = (S_c, E, I, A, T)$ (susceptible, exposed, symptomatic-infectious, asymptomatic-infectious, treatment), with ten rate scalars in the compartmental drift and an Atangana-Seda quadratic-in-state diffusion driven by five independent Brownian motions; to avoid clashes with our notation, we write α_T for Bonyah's treatment-recovery rate (their α) and χ for the fractional order (our α). We compare three configurations, one mechanistic and two free-form, against a naive baseline. **Parametric FSDE**: Bonyah's compartmental drift with the ten rate scalars learnable, the Atangana-Seda diffusion with the ten noise intensities $\Lambda_{j,k}$ learnable, and learnable χ; this is the reference Bonyah FSDE inside our adjoint framework. **Neural FSDE**: a free-form 3-D latent $X \in \mathbb{R}^3$ with a neural drift $\mu_\theta(X)$, neural diffusion $\sigma_\phi(X)$, and exponential observation map $(\text{cases}, \text{deaths}) = (\exp X_0, \exp X_1)$ (subscripts index latent coordinates, not time); the first two coordinates start at the log of the window-start observations and X_2 is a hidden state with a learnable initial value; χ is learnable. **Neural SDE**: same architecture as the Neural FSDE but with $\chi \equiv 1$ frozen; the shared 3-D latent architecture isolates the effect of fractional memory. **Persistence**: the last observation carried forward, $\hat{y}_{t_0+h} = y_{t_0}$ for forecast origin t_0 and horizon h. On paths simulated from the Bonyah model with known order $\chi^* \in \{0.6, 0.75, 0.9\}$ (Appendix C.3), the Parametric FSDE and a hybrid with a neural diffusion recover $\hat{\chi}$ to within 0.01 of χ^*. A variant with a neural drift on Bonyah's 5-D state, which is distinct from the 3-D latent models above, fails to identify the order, with $\hat{\chi} \to 1$ whatever χ^*, consistent with a flexible neural drift using the compartments as auxiliary states to emulate memory. On real data we therefore read the recovered $\hat{\chi}$ off the Parametric FSDE. Network architecture and hyperparameters in Appendix C.4.

Data. We calibrate to daily records from Our World in Data (OWID, `https://covid.ourworldindata.org`) on four slices, each a window of $\sim$181–184 days around one epidemic wave in one country (UK Alpha, UK Delta, US Winter, US Delta); the Delta windows end in a rising tail, and the Alpha and Winter windows include the decline after a peak. The Parametric FSDE retains Bonyah's observation map from the 5-D state (in Bonyah's symbols, cases $= \beta S_c(I + \theta_1 T + \theta_2 A)$ and deaths $= \mu(I + A + T)$, with transmission rate β, infectiousness modifiers θ_1, θ_2 and mortality rate μ); the Neural FSDE and Neural SDE use the exponential 3-D map. *Rolling-origin protocol*: at each forecast origin, day $t_0 \in \{60, 67, \ldots, 165\}$ of the slice (16 weekly origins per slice), we calibrate on the preceding $N = 60$-day window, holding out its last 7 days to select the best checkpoint, and forecast $h = 7$ days ahead. Each calibration is repeated under three random seeds, giving 48 squared errors per (slice, channel) cell, where the channel is cases or deaths. Forecasts are persistence-anchored

deltas $\hat{y}_{t_0+h} = y_{t_0} + (\mathrm{model}_{t_0+h} - \mathrm{model}_{t_0})$, where model_t is the model's predicted observation on day t, averaged over its simulated paths. Bonyah et al. [10] only fitted SARIMA to South African counts and never calibrated the FSDE itself; ours is the first such end-to-end calibration.

Results. Table 1 reports the median 7-day-ahead RMSE, that is, the square root of the median squared error in each cell; rolling overlays (Figure 8), per-cell error distributions (Figure 7), and recovered Bonyah parameters are in Appendix C.1–C.2. One of the two FSDE variants (Parametric FSDE and Neural FSDE) is best in five of eight cells: the Parametric FSDE on UK Delta cases and on the deaths channel of UK Alpha and US Winter, and the Neural FSDE on UK Alpha and US Winter cases. The memoryless Neural SDE is best on both channels of US Delta, and Persistence on UK Delta deaths. Against the memoryless Neural SDE, with the same 3-D architecture and seeds, the Neural FSDE is more accurate in five of eight cells, including three of four on the more volatile cases channel. Two of the 192 Neural SDE calibrations produced anomalously large forecasts (UK Alpha origin 165, US Winter origin 130), and none of the Neural FSDE's did under the same architecture and seeds. This is consistent with the Caputo weights damping the accumulated noise at the daily step, where the first Caputo noise weight is about 0.75 times the memoryless one and later weights decay (Appendix D.4). The Parametric FSDE recovers $\hat{\chi} < 1$ on all 192 calibrations (per-slice medians $\hat{\chi} \in [0.67, 0.71]$, full range $[0.58, 0.81]$), supporting the long-memory regime [10] previously proposed but not empirically calibrated. The per-cell differences in Table 1 are directional rather than individually significant: Diebold-Mariano tests of equal forecast accuracy with a small-sample correction [17, 31], applied to the 16 seed-averaged weekly origins, find no difference significant at 5% for any pairing that involves an FSDE variant (minimum $p = 0.057$; Appendix C.1). The same appendix adds seven baselines under the identical protocol. Five are fitted on each window: a neural controlled differential equation (Neural CDE), a Neural SVE, the classical long-memory model ARFIMA (autoregressive fractionally integrated moving average), an LSTM and a causal Transformer. Two are pretrained zero-shot forecasters, Chronos and TimesFM, reported out of competition because their public training corpora plausibly contain COVID-19 series. No method wins uniformly, and none of the seven is more accurate than the Neural FSDE in any cell under both the Diebold-Mariano test and an origin-clustered bootstrap. ARFIMA is the strongest of the baselines fitted on each window and beats the Parametric FSDE on UK Alpha cases under the Diebold-Mariano test.

Table 1: Median 7-day-ahead forecast RMSE on the four OWID slices under the rolling-origin protocol (16 weekly origins $\times$ 3 seeds $= 48$ squared errors per cell): cases per million per day, with deaths per million per day in parentheses. Lower is better; bold = per-channel row-best.

Slice	Parametric FSDE	Neural FSDE	Neural SDE	Persistence
United Kingdom (Alpha)	104.7 (**0.75**)	**74.4** (1.02)	81.1 (1.25)	75.2 (1.23)
United Kingdom (Delta)	**49.2** (0.23)	63.0 (0.19)	72.0 (0.18)	60.3 (**0.16**)
United States (Winter)	75.6 (**0.36**)	**57.6** (0.41)	66.5 (0.48)	74.8 (0.52)
United States (Delta)	47.3 (0.63)	34.3 (0.67)	**31.8** (**0.51**)	33.0 (0.53)

4.3 Option Pricing

Setup. We apply our framework to S&P 500 index option pricing under stochastic volatility, in two parts: *European* pricing (exercise at expiry) on SPX weekly options (SPXW), and *American* pricing (exercise at any time) on SPDR S&P 500 ETF (SPY) options. The spot S_t with instantaneous variance V_t follows $\mathrm{d}S_t/S_t = \sqrt{V_t}\,\mathrm{d}Z_t$, $\mathrm{d}Z_t = \rho\,\mathrm{d}W_t + \sqrt{1-\rho^2}\,\mathrm{d}W_t^{\perp}$, for independent Brownian motions W and $W^{\perp}$ and correlation ρ, with W also driving the variance process. We compare four model classes, all recast as Caputo FSDEs with $\alpha \in (\frac{1}{2}, 1]$ so one Caputo discretisation serves all four. **rough Bergomi** [5] sets $V_t = \xi_0(t)\,\mathcal{E}(\eta\,\tilde{W}_t^H)$, with volatility of volatility η, via the Wick exponential of a Volterra integral $\tilde{W}_t^H = \sqrt{2H}\int_0^t (t-s)^{H-1/2}\,\mathrm{d}W_s$; in our framework, it is a Caputo FSDE on $L_t = \log V_t$ with constant diffusion intensity $\eta\sqrt{2H}\,\Gamma(\alpha)$, $\alpha = H + \frac{1}{2}$, and a learnable 4-knot piecewise-constant log forward variance curve $\xi_0(u)$ that absorbs the deterministic Wick correction $-\frac{1}{2}\eta^2 t^{2H}$. **rough Heston** [22] sets $V_t = V_0 + \frac{1}{\Gamma(\alpha)}\int_0^t (t-s)^{\alpha-1}[\lambda(\theta - V_s)\,\mathrm{d}s + \lambda\nu\sqrt{V_s}\,\mathrm{d}W_s]$, exactly a Caputo FSDE on V with mean-reverting drift (speed λ, level θ, volatility of variance ν) and a square-root (CIR) diffusion with full truncation. **Neural SDE** ($\alpha \equiv 1$) shares the drift and diffusion of the Neural FSDE below but freezes $\alpha = 1$ to isolate the contribution of memory at fixed

architecture. **Neural FSDE** (ours) is a Caputo FSDE on $L_t = \log V_t$ with neural diffusion σ_ϕ and drift $\mu_\theta(L_t) + \lambda\,(\log \xi_0(t) - L_t)$, a small neural correction μ_θ added to a mean reversion towards a learnable 4-knot piecewise-constant log forward variance curve $\log \xi_0(t)$ as in rough Bergomi. Both nets are 3-layer time-invariant MLPs (hidden 64, $\tanh$-bounded σ_ϕ output), λ is a learnable scalar mean-reversion rate, the four $\xi_0(\cdot)$ knots are initialised per day from the squared at-the-money (ATM) implied volatility IV^2 in each of four maturity bins (edges at 30, 90 and 180 days; Appendix D.5), and the fractional order is learnable via $\alpha = \frac{1}{2} + \frac{1}{2}\,\mathrm{sigmoid}(u_\alpha)$. Both neural models bound their diffusion, $\sigma_\phi = c\tanh(\cdot)$. Because on the option grid ($\Delta t = 1/128$ year) the Caputo noise prefactor grows as α falls (Eq. 9), the memoryless model needs a larger diffusion for the same one-step noise, and a shared bound would handicap it. Each neural model therefore uses its own bound, the value $c \in \{3, 6, 12, 24\}$ with the lowest median error on pilot subsets of 61 trading days: $c = 3$ for the Neural FSDE and $c = 12$ for the Neural SDE (Appendix D.4). The $\log V$ parametrisation keeps $V > 0$ a.s.; rough Heston works in V-space with full truncation per its standard convention. As an external reference outside the Caputo family, we consider **PDV**, the four-factor path-dependent volatility model [30], whose volatility is an explicit function of exponentially weighted averages of past returns and squared returns; its nine scalars are recalibrated per day under the same protocol, with the averages at $t = 0$ built from trailing S&P 500 returns (Appendix D.2).

Data. We use end-of-day option chains over 2024-03-11–2026-03-31 (483 trading days; SPXW has 550 expirations, SPY has maturities up to $\sim$3 years). After standard filters on time to maturity $\tau \in [7\mathrm{d}, 365\mathrm{d}]$, implied volatility (IV) $\in [0.03, 1.5]$, log-moneyness $k = \log(K/F)$ (strike K, forward F) with $|k| \le 0.30$, and price $\ge \$0.10$, keeping out-of-the-money quotes only, the SPXW set has 1,693,453 quotes and SPY has 896,182 quotes. We split each implied-volatility surface into four quadrants by maturity, short ($\tau \le 60$d) or long, and moneyness, near-money ($|k| \le 0.05$) or far-money: *short, near-money*; *short, far-money*; *long, near-money*; and *long, far-money*.

Protocol. For each (date, model) pair we calibrate from a fresh cold-start initialisation, not from the previous day's fit, under a quadrant-balanced log-price MSE applied uniformly to all models, $\sum_{q\in\mathcal{Q}} \frac{1}{|q|} \sum_{i\in q} (\log(C_i^{\text{model}} + \varepsilon) - \log(C_i^{\text{market}} + \varepsilon))^2$, where $\mathcal{Q}$ is the set of quadrants, C_i^{model} and C_i^{market} are the model and market prices of quote i, and $\varepsilon = 0.01$. We use log prices because they weight relative errors, which up-weights the small prices of far-money quotes (the wings of the smile), whereas a vega-weighted price MSE (vega being the price sensitivity to volatility) relies on a first-order expansion of price in volatility that breaks down for these quotes. Quadrant balancing keeps the near-money body of the surface from absorbing all the gradient signal. For evaluation, each model price, computed from 32,768 Monte Carlo paths, is converted to an implied volatility by Newton inversion of the European Black-Scholes formula (SPXW) or the Bjerksund-Stensland 2002 American approximation [9] (SPY), and we report the root-mean-square difference from the market IV (IV-RMSE) in basis points of volatility. For SPY, dividends and carry enter through a per-day forward curve fitted to the quoted forwards. Early exercise enters through least-squares Monte Carlo on a degree-2 polynomial basis in $(S_t, V_t, V_{t-1}, V_{t-2}, V_{t-3})$, with lags in time steps; all classes share this pricer, which is differentiated with the exercise policy held fixed. Full data attributes, training hyperparameters, and the pricing and evaluation details are in Appendix D.2.

We report four evaluation schedules per instrument: **(a)** in-sample fit per day; **(b)** day-ahead out-of-sample (OOS) pricing of day $t{+}1$ quotes with day t's parameters, without re-calibration, which bounds from above the pricing error from parameter drift over a typical trading-desk recalibration cycle; **(c)** expiry-stratified holdout (one expiry held out in each of three maturity ranges $\{\tau \le 30\mathrm{d},\ 30\text{–}180\mathrm{d},\ >180\mathrm{d}\}$, the same for all models, evaluated on the held-out expiries); **(d)** liquidity-stratified holdout (calibration on the top 30% of quotes by daily volume and testing on the rest). For (d), the calibration loss is balanced over the four maturity bins of the ξ_0 knots, since the volume-skewed training set would otherwise dilute the long-maturity training signal. Schedule (d) mirrors practice, where a model is calibrated on liquid quotes and then used to price illiquid ones.

Results. Table 2 reports the median IV-RMSE for both instruments under the four schedules, pooled over all of a day's quotes. Under our unified cold-start calibration pipeline the Neural FSDE attains the lowest median IV-RMSE on all eight settings, with margins of 19–128 bps over rough Heston (the strongest baseline) and 87–255 bps over PDV. All 32 paired comparisons of the Neural FSDE with a baseline have 95% bootstrap intervals below zero (Appendix D.1). Raising the calibration budget from 60 to 200 Adam steps for rough Bergomi, rough Heston and the Neural FSDE keeps the Neural FSDE below both classical models on every setting and widens its margin on seven of eight (Appendix D.3). Per-quadrant tables, boxplots, and smile fits are in Appendix D.1.

Memory and diffusion scale. The Neural SDE row differs from the Neural FSDE only in α and in the diffusion bound. Because each model selects its own bound by the same rule, the remaining difference isolates the fractional order. The bound matters because the Caputo prefactor $\Delta t^{\alpha-1/2}/\Gamma(\alpha)$ makes the one-step noise about four times that of the memoryless model at $\hat{\alpha} \approx 0.64$ for the same diffusion. With both models at the common bound $c = 3$, the Neural SDE's diffusion sits at the bound on 99.2% of visited states, and raising the Neural SDE's bound to its best value, $c = 12$, removes 60–74% of the Neural FSDE's lead at $c = 3$ (Appendix D.4). With each model at its best bound, the fractional order still lowers the median IV-RMSE on all eight settings, by 69–132 bps. Every 95% bootstrap interval of this difference lies at least 60 bps below zero, and the Neural FSDE is lower on 435–480 of about 483 days per setting. At the common bound $c = 12$, which is the Neural SDE's best but not the Neural FSDE's, the Neural FSDE still leads on all eight settings, by 23–88 bps (Appendix D.4). Schedule (b) re-prices day t's calibration, so only the six settings of schedules (a), (c) and (d) have calibrations of their own. On these, per-day $\hat{\alpha}$ has per-setting medians 0.638–0.675 and exceeds 0.95 on none of the 2,897 calibrations. Under $H = \alpha - \frac{1}{2}$ this is a Hurst index near 0.15, of the order of the $H \approx 0.1$ estimated for equity volatility [24] and rougher than the rough-Bergomi and rough-Heston fits in the same pipeline ($H \approx 0.20$; Figure 14). With α frozen, the error rises steeply as the frozen value approaches 1 and has a shallow basin on $[0.55, 0.65]$, where the learnable-α model comes within 6–9 bps of the best fixed value (Appendix D.6).

Table 2: Median IV-RMSE (bps) across 483 trading days, all quotes, on SPXW (European) and SPY (American) under the four schedules: (a) in-sample, (b) day-ahead OOS, (c) expiry-stratified, (d) liquidity-stratified with a maturity-balanced loss. PDV is the path-dependent volatility model [30]. Each neural model uses its best diffusion bound (Appendix D.4). Bold indicates the best performance per column.

	SPXW (European)				SPY (American)			
Model	(a)	(b)	(c)	(d)	(a)	(b)	(c)	(d)
rough Bergomi	585	727	407	537	444	448	384	508
rough Heston	344	347	232	296	211	223	207	206
PDV	450	524	388	391	263	283	261	292
Neural SDE	348	382	285	328	244	260	244	278
Neural FSDE	**216**	**269**	**212**	**212**	**165**	**182**	**175**	**187**

Residual structure. Per quadrant (Tables 13 and 14), on SPXW the Neural FSDE's gain over rough Heston is concentrated in the steep *short, far-money* weekly wing (280 vs. 515 bps under (a)), where the neural diffusion is more flexible than rough Heston's square-root (CIR) diffusion. The Neural FSDE's largest deficit is on the held-out *long, near-money* expiry under (c) (263 vs. 98 bps), where rough Heston's six scalars transfer better across maturities. On SPY the Neural FSDE leads at short maturities except in the *short, far-money* quadrant under (d) (8 bps behind rough Heston); at long maturities it is 5–35 bps behind rough Heston under (a)–(c) and 18–32 bps ahead under (d).

5 Conclusion

We introduced the Neural Fractional Stochastic Differential Equation together with a discrete adjoint method that computes gradients with respect to the initial state, the drift and diffusion parameters, and the fractional order α in $\mathcal{O}(N \cdot D \cdot B)$ memory. We validated the framework on two real-world problems: an end-to-end real-data calibration of Bonyah's stochastic-fractional COVID-19 model on OWID data, and an S&P 500 option-pricing benchmark on which the Neural FSDE attains the lowest median IV-RMSE on both SPXW and SPY options. Against an architecture-matched memoryless Neural SDE given its best diffusion bound, the fractional order lowers the error by 69–132 bps; when both models share the diffusion bound $c = 3$, about two-thirds of the gap at that bound is a diffusion-scale effect, because α also scales the effective volatility of volatility. Directions for future work are extending the adjoint to numerical schemes beyond the Euler-Maruyama discretisation used here, and lifting the framework to other FSDE forms (e.g., Atangana-Baleanu kernels) and to the more general stochastic Volterra equations of which Caputo FSDEs are a special case.

Disclaimer: All authors are faculty. Neither graduate students nor small animals were hurt while producing this paper.

References

[1] E. Abi Jaber and O. El Euch. Multifactor approximation of rough volatility models. *SIAM Journal on Financial Mathematics*, 10(2):309–349, 2019.

[2] E. Abi Jaber, M. Larsson, and S. Pulido. Affine Volterra processes. *The Annals of Applied Probability*, 29(5):3155–3200, 2019.

[3] A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. S. Rangapuram, S. Pineda Arango, S. Kapoor, J. Zschiegner, D. C. Maddix, H. Wang, M. W. Mahoney, K. Torkkola, A. G. Wilson, M. Bohlke-Schneider, and Y. Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024.

[4] F. Baschetti, G. Bormetti, and P. Rossi. Deep calibration with random grids. *Quantitative Finance*, 24(9):1263–1285, 2024.

[5] C. Bayer, P. Friz, and J. Gatheral. Pricing under rough volatility. *Quantitative Finance*, 16(6): 887–904, 2016.

[6] C. Bayer, B. Horvath, A. Muguruza, B. Stemper, and M. Tomas. On deep calibration of (rough) stochastic volatility models. *The Journal of FinTech*, 2025. doi: 10.1142/S2705109925500051.

[7] M. Bennedsen, A. Lunde, and M. S. Pakkanen. Hybrid scheme for Brownian semistationary processes. *Finance and Stochastics*, 21(4):931–965, 2017.

[8] M. Bergerhausen, D. J. Prömel, and D. Scheffels. Neural stochastic Volterra equations: Learning path-dependent dynamics. *Journal of Machine Learning*, 4(4):264–289, 2025.

[9] P. Bjerksund and G. Stensland. Closed form valuation of American options. Discussion paper, NHH Norwegian School of Economics, 2002.

[10] E. Bonyah, M. L. Juga, L. M. Matsebula, and C. W. Chukwu. On the modeling of COVID-19 spread via fractional derivative: A stochastic approach. *Mathematical Models and Computer Simulations*, 15(2):338–356, 2023.

[11] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

[12] C. Coelho, M. F. P. Costa, and L. L. Ferrás. Neural fractional differential equations. *Applied Mathematical Modelling*, 144:116060, 2025.

[13] W. Cui, Q. Kang, X. Li, K. Zhao, W. P. Tay, W. Deng, and Y. Li. Neural variable-order fractional differential equation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[14] R. Daems, M. Opper, G. Crevecoeur, and T. Birdal. Variational inference for SDEs driven by fractional noise. In *International Conference on Learning Representations*, 2024.

[15] A. Das, W. Kong, R. Sen, and Y. Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10148–10167, 2024.

[16] D. Denu and S. Kermausuor. Analysis of a fractional-order COVID-19 epidemic model with lockdown. *Vaccines*, 10(11):1773, 2022.

[17] F. X. Diebold and R. S. Mariano. Comparing predictive accuracy. *Journal of Business & Economic Statistics*, 13(3):253–263, 1995.

[18] K. Diethelm, N. J. Ford, and A. D. Freed. A predictor-corrector approach for the numerical solution of fractional differential equations. *Nonlinear Dynamics*, 29(1–4):3–22, 2002.

[19] T. S. Doan, P. T. Huong, P. E. Kloeden, and A. M. Vu. Euler–Maruyama scheme for Caputo stochastic fractional differential equations. *Journal of Computational and Applied Mathematics*, 380:112989, 2020.

[20] L. dos Santos Lima. Fractional stochastic differential equation approach for spreading of diseases. *Entropy*, 24(5):719, 2022.

[21] E. Dupont, A. Doucet, and Y. W. Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, 2019.

[22] O. El Euch and M. Rosenbaum. The characteristic function of rough Heston models. *Mathematical Finance*, 29(1):3–38, 2019.

[23] T. Gaskin, T. O. F. Conrad, G. A. Pavliotis, and C. Schütte. Neural parameter calibration and uncertainty quantification for epidemic forecasting. *PLOS ONE*, 19(10):e0306704, 2024. doi: 10.1371/journal.pone.0306704.

[24] J. Gatheral, T. Jaisson, and M. Rosenbaum. Volatility is rough. *Quantitative Finance*, 18(6): 933–949, 2018.

[25] P. Gierjatowicz, M. Sabate-Vidales, D. Šiška, Ł. Szpruch, and Ž. Žurič. Robust pricing and hedging via neural stochastic differential equations. *Journal of Computational Finance*, 26(3): 1–32, 2022. doi: 10.21314/JCF.2022.025.

[26] P. Glasserman. *Monte Carlo Methods in Financial Engineering*, volume 53 of *Stochastic Modelling and Applied Probability*. Springer, New York, 2003.

[27] C. W. J. Granger and R. Joyeux. An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis*, 1(1):15–29, 1980.

[28] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

[29] A. Gu, K. Goel, and C. Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

[30] J. Guyon and J. Lekeufack. Volatility is (mostly) path-dependent. *Quantitative Finance*, 23(9): 1221–1258, 2023.

[31] D. Harvey, S. Leybourne, and P. Newbold. Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, 13(2):281–291, 1997.

[32] K. Hayashi and K. Nakagawa. Fractional SDE-net: Generation of time series data with long-term memory. In *International Conference on Data Science and Advanced Analytics*, 2022.

[33] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

[34] B. Horvath, A. Muguruza, and M. Tomas. Deep learning volatility: A deep neural network perspective on pricing and calibration in (rough) volatility models. *Quantitative Finance*, 21(1): 11–27, 2021.

[35] J. R. M. Hosking. Fractional differencing. *Biometrika*, 68(1):165–176, 1981.

[36] S. Jiang, J. Zhang, Q. Zhang, and Z. Zhang. Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations. *Communications in Computational Physics*, 21(3):650–678, 2017.

[37] Q. Kang, K. Zhao, Q. Ding, F. Ji, X. Li, W. Liang, Y. Song, and W. P. Tay. Unleashing the potential of fractional calculus in graph neural networks with FROND. In *International Conference on Learning Representations*, 2024.

[38] Q. Kang, X. Li, K. Zhao, W. Cui, Y. Zhao, W. Deng, and W. P. Tay. Efficient training of neural fractional-order differential equation via adjoint backpropagation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[39] P. Kidger, J. Morrill, J. Foster, and T. Lyons. Neural controlled differential equations for irregular time series. In *Advances in Neural Information Processing Systems*, 2020.

[40] P. Kidger, J. Foster, X. Li, and T. Lyons. Efficient and accurate gradients for neural SDEs. In *Advances in Neural Information Processing Systems*, 2021.

[41] P. Kidger, J. Foster, X. Li, H. Oberhauser, and T. Lyons. Neural SDEs as infinite-dimensional GANs. In *International Conference on Machine Learning*, 2021.

[42] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. K. Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[43] F. A. Longstaff and E. S. Schwartz. Valuing American options by simulation: A simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147, 2001.

[44] Y. Luchko and M. Yamamoto. The general fractional derivative and related fractional differential equations. *Mathematics*, 8(12):2115, 2020.

[45] S. McCallum and J. Foster. Efficient, accurate and stable gradients for neural ODEs. *arXiv preprint arXiv:2410.11648*, 2024.

[46] G. Nobis, M. Springenberg, M. Aversa, M. Detzel, R. Daems, R. Murray-Smith, S. Nakajima, S. Lapuschkin, S. Ermon, T. Birdal, M. Opper, C. Knochenhauer, L. Oala, and W. Samek. Generative fractional diffusion models. In *Advances in Neural Information Processing Systems*, 2024.

[47] Y. Oh, D. Lim, and S. Kim. Stable neural stochastic differential equations in analyzing irregular time series data. In *International Conference on Learning Representations*, 2024.

[48] G. Pang, L. Lu, and G. E. Karniadakis. fPINNs: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4):A2603–A2626, 2019.

[49] P. M. Robinson. Gaussian semiparametric estimation of long range dependence. *The Annals of Statistics*, 23(5):1630–1661, 1995.

[50] Y. Rubanova, R. T. Q. Chen, and D. K. Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, 2019.

[51] D. Shmelev, L. Thompson, and C. Salvi. Explicit and effectively symmetric schemes for neural SDEs on Lie groups. *arXiv preprint arXiv:2509.20599*, 2026.

[52] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[53] A. Tong, T. Nguyen-Tang, T. Tran, and J. Choi. Learning fractional white noises in neural stochastic differential equations. In *Advances in Neural Information Processing Systems*, 2022.

[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[55] J. Zhang, Y. Tang, and J. Huang. A fast Euler–Maruyama method for fractional stochastic differential equations. *Journal of Applied Mathematics and Computing*, 69(1):273–291, 2023.

[56] J. Zhuang, N. Dvornek, X. Li, S. Tatikonda, X. Papademetris, and J. Duncan. Adaptive checkpoint adjoint method for gradient estimation in neural ODE. In *International Conference on Machine Learning*, 2020.

[57] B. Zimmering, C. Coelho, and O. Niggemann. Optimising neural fractional differential equations for performance and efficiency. In *Proceedings of the 1st ECAI Workshop on Machine Learning Meets Differential Equations: From Theory to Applications*, pages 1–22, 2024.

[58] K. Zubov, Z. McCarthy, Y. Ma, F. Calisto, V. Pagliarino, S. Azeglio, L. Bottero, E. Luján, V. Sulzer, A. Bharambe, et al. NeuralPDE: Automating physics-informed neural networks (PINNs) with error approximations. *arXiv preprint arXiv:2107.09443*, 2021.

A Adjoint vs. Autograd Benchmark Details

We benchmark our discrete adjoint against direct PyTorch backpropagation through the Euler-Maruyama solver. Both methods execute the same forward Euler-Maruyama (EM) scheme on a single NVIDIA H100 GPU; only the backward pass differs. The drift and diffusion are 3-layer MLPs with $\tanh$ activations. For the sweeps of Figures 2–4 we report the median wall-clock time over 3 timed iterations after a warm-up, and peak GPU memory for the forward+backward step. Autograd memory grows steeply with the time grid because every intermediate Y_n, μ_n, σ_n and the entire pairwise weight tensor must be retained for the backward pass, whereas the adjoint only stores the trajectory and noise samples. Summary numbers are quoted in Section 4.1; the three sweep figures below show how the savings scale with N, B, and H separately.

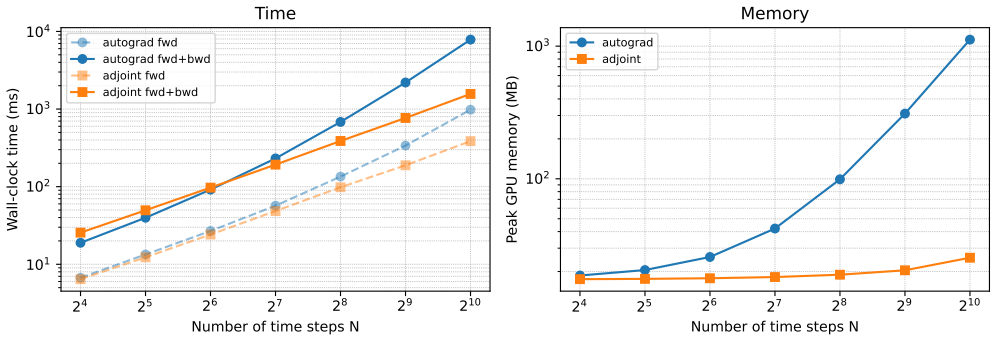


Figure 2: Adjoint vs. autograd, sweeping the number of time steps N at B=64, H=64. Left: wall-clock time. Right: peak GPU memory. Both axes log-scaled.

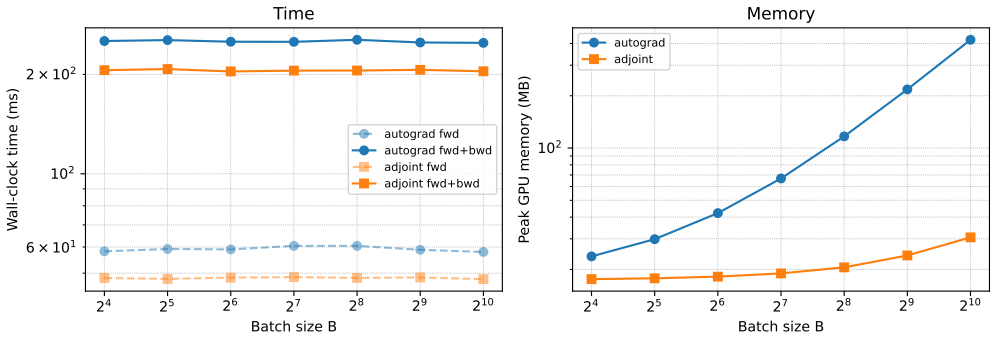


Figure 3: Adjoint vs. autograd, sweeping the number of paths B at N=128, H=64.

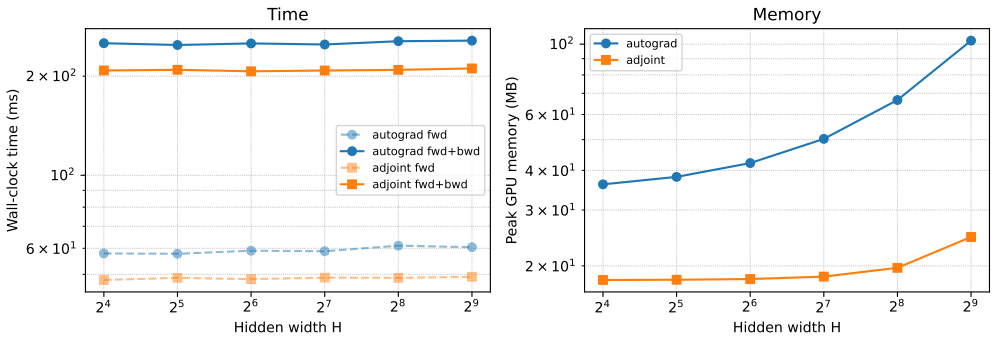


Figure 4: Adjoint vs. autograd, sweeping the hidden width H at N=128, B=64.

Table 3 reports the forward and backward wall-clock times separately at the canonical sweep points (median of five iterations after two warm-up steps). Both methods run the same Euler-Maruyama forward, so on the small base point (N=128, B=64, H=64) their forward timings agree to within

about 20% (48.1 vs. 57.8 ms). Savings in both time and memory appear once we scale N: from N=128 to N=1024 the adjoint's backward grows from 148 to 1,166 ms, while autograd's rises to 6,833 ms, a 5.9× backward speed-up at the same point where autograd's peak memory is 44 times the adjoint's (1,119.8 vs. 25.4 MB). Scaling B or H at fixed N=128 leaves wall-clock close to the base point but shifts memory proportionally, since the autograd graph stores tensors of shape (B, N, D) for every intermediate.

Table 3: Adjoint vs. autograd: forward and backward wall-clock split at four canonical sweep points (median of five iterations after two warm-up steps). The backward speed-up scales with N, where the autograd graph traversal cost is paid; the per-step forward cost is dominated by the same Euler-Maruyama loop both methods execute.

	Autograd			Adjoint			
Setting	fwd (ms)	bwd (ms)	tot (ms)	fwd (ms)	bwd (ms)	tot (ms)	bwd speed-up
N=128, B=64, H=64 (base)	57.8	177.4	235.2	48.1	147.8	195.9	1.2×
N=1024, B=64, H=64	974.7	6832.7	7807.3	384.1	1165.7	1549.9	5.9×
N=128, B=1024, H=64	59.6	178.4	237.9	48.8	149.4	198.2	1.2×
N=128, B=64, H=512	59.5	179.8	239.3	49.0	150.0	199.0	1.2×

A.1 Toy α-Initialisation Sensitivity

To verify that the recovery $\hat{\alpha} = 0.7516$ in Section 4.1 is not specific to its initialisation $\alpha = 0.55$, we re-ran the toy training, on one shared training sample drawn independently of that of Section 4.1, at five α-initialisations spanning the full $(0.5, 1.0]$ admissible range, with three random seeds per initialisation (15 runs, 2,000 training steps each). Across all 15 runs the recovered $\hat{\alpha}$ (averaged over the last 100 steps) falls in $[0.7537, 0.7684]$ around the truth $\alpha^* = 0.75$, with per-init mean $\hat{\alpha} \in \{0.7572, 0.7597, 0.7618, 0.7636, 0.7642\}$ for $\alpha_{\text{init}} \in \{0.55, 0.65, 0.75, 0.85, 0.95\}$ respectively (Table 4, Figure 5). Within-initialisation seed std is 0.0022–0.0041, comparable to the across-initialisation spread, so neither seed noise nor initialisation bias dominates the recovered estimate.

Table 4: Toy sanity check: recovered fractional order $\hat{\alpha}$ as a function of initialisation, three seeds per setting, 2,000 training steps, truth $\alpha^* = 0.75$. The final $\hat{\alpha}$ is the average over the last 100 training steps. Recovery holds for every initialisation within ± 0.20 of α^*.

α_{init}	$\hat{\alpha}$ mean (3 seeds)	$\hat{\alpha}$ std	$\hat{\alpha}$ range	final loss
0.55	0.7572	0.0035	[0.7537, 0.7608]	0.0171
0.65	0.7597	0.0026	[0.7578, 0.7627]	0.0174
0.75	0.7618	0.0022	[0.7594, 0.7638]	0.0175
0.85	0.7636	0.0032	[0.7601, 0.7662]	0.0176
0.95	0.7642	0.0041	[0.7602, 0.7684]	0.0175

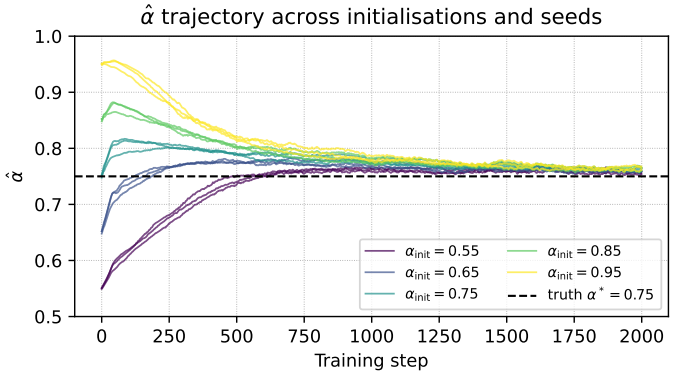


Figure 5: $\hat{\alpha}$ trajectory across five initialisations and three seeds on the toy sanity check (truth $\alpha^* = 0.75$). All 15 runs converge into a narrow band $[0.7537, 0.7684]$ around the truth.

B Numerical Analysis and a Fast Kernel

This appendix collects further numerical checks: the cost of each solver mode (B.1), the strong convergence of the Euler-Maruyama scheme (B.2), robustness to a predictor-corrector forward (B.3), a sum-of-exponentials fast kernel (B.4), and adjoint gradients near the lower edge $\alpha \to \frac{1}{2}$ (B.5).

B.1 Cost Summary

A Neural SDE's Euler-Maruyama step costs $\mathcal{O}(1)$, so its forward pass costs $\mathcal{O}(N)$. The Caputo step of Eq. 10 must also contract every stored drift and diffusion evaluation with the kernel weights over the whole history, so the forward pass costs $\mathcal{O}(N^2)$. Table 5 lists the cost of each mode. The adjoint's benefit is entirely on the backward side: it removes the $\mathcal{O}(N^2)$ autograd graph and the cost of traversing it, while its own backward recursion keeps $\mathcal{O}(N^2)$ time and $\mathcal{O}(NBD)$ memory (B paths, D state dimensions), a $44\times$ peak-memory reduction and a $5.9\times$ backward speed-up at $N = 1024$ (Appendix A).

Table 5: Cost of one training step with the exact Caputo kernel and with a K-term sum-of-exponentials (SOE) approximation (Appendix B.4), as a function of the number of steps N. Every entry also scales linearly with the number of paths B and the state dimension D; these factors are written out only in the adjoint-memory row, where NBD is the whole cost. Neither of the two SOE adjoints in the table has been implemented.

	Exact kernel	Sum-of-exponentials (K terms)
Forward time	$\mathcal{O}(N^2)$	$\mathcal{O}(NK)$
Backward time, autograd	$\mathcal{O}(N^2)$	$\mathcal{O}(NK)$
Backward time, adjoint	$\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$ reusing ours; $\mathcal{O}(NK)$ if recursion-aware
Backward memory, autograd	$\mathcal{O}(N^2)$	$\mathcal{O}(NK)$
Backward memory, adjoint	$\mathcal{O}(NBD)$	$\mathcal{O}(NBD)$

B.2 Strong Convergence of the Euler-Maruyama Scheme

The strong-convergence theory for the Euler-Maruyama discretisation of Caputo FSDEs is due to Doan et al. [19], with an error constant that degrades as α approaches $\frac{1}{2}$ through the $\mathcal{O}(1/(2\alpha - 1))$ variance term of the left-point diffusion rule (Section 3.2). We check that our implementation shows this behaviour on $D^\alpha X = \cos X + \sin X\, \dot{W}$, $X_0 = 0$, $T = 1$, in double precision. A reference solution uses $N_{\text{fine}} = 8192$ steps; each coarse solve with $N \in \{64, \dots, 2048\}$ reuses the same Brownian path, coarsened by summation, and the strong error $\mathbb{E}|X_N(T) - X_{\text{fine}}(T)|$ is averaged over $B = 4096$ paths (Figure 6). We bracket the strong order by two fits: the naive log-log slope of error against step size is an upper bracket, because the reference itself carries error, and the fit $C(\Delta t^p - \Delta t_{\text{fine}}^p)$ is a lower bracket, because it treats the errors at different resolutions as fully correlated. Without noise ($\sigma \equiv 0$) the scheme converges at order 1.00–1.08 for every α. With noise the brackets are $[0.00, 0.19]$, $[0.00, 0.24]$, $[0.02, 0.36]$ and $[0.21, 0.47]$ for $\alpha = 0.55$, 0.6, 0.75 and 0.9, each containing the rate $\alpha - \frac{1}{2}$, and the error at $N = 1024$ grows about twentyfold from $\alpha = 0.9$ (0.011) to $\alpha = 0.55$ (0.214), the degradation the theory predicts.

B.3 Predictor-Corrector Forward

Our discrete adjoint differentiates the recursion of Eq. 8. To check that our conclusions do not hinge on that scheme, we implemented the Adams-type predictor-corrector (PECE) of Diethelm et al. [18] for the drift, keeping the left-point rule for the diffusion, and trained through it with autograd as the backward pass, which is feasible at synthetic scale. We make three checks on the toy system of Section 4.1. (i) On a shared fine noise path ($N = 4096$, $B = 4096$) the terminal Wasserstein-1 distance between the EM and PECE solutions is 3.7×10^{-4}, 7.2×10^{-5} and 3.8×10^{-5} for $\alpha = 0.6$, 0.75 and 0.9, between 96 and 384 times below the Monte Carlo floor, the distance between two runs of the same solver with independent noise. (ii) At our operating resolution $N = 100$, the two schemes' time-averaged distances to a fine PECE reference differ by at most 4% (0.073 vs. 0.075, 0.021 vs. 0.022, 0.0077 vs. 0.0075), because the dominant coarse-grid error is the diffusion rule they share. (iii) Training through the PECE forward recovers $\hat{\alpha} = 0.754$, against 0.7516 for EM with our

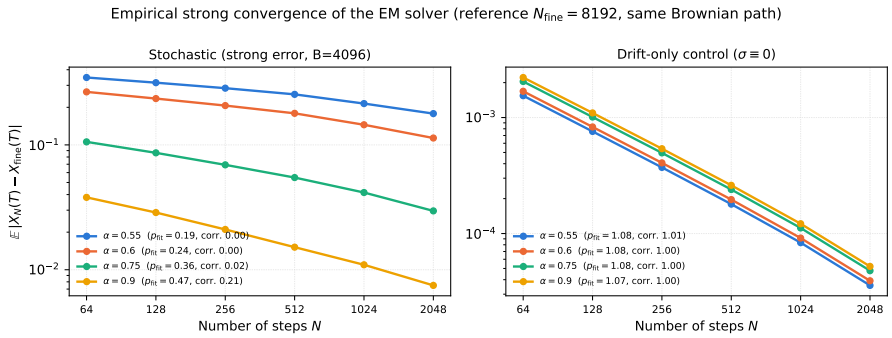


Figure 6: Strong error of the Euler-Maruyama solver against a common-path reference with $N_{\text{fine}} = 8192$. Left: with noise. Right: drift only. Legends give the naive and reference-corrected fitted orders.

adjoint (truth 0.75). An adjoint for PECE would also need the chain rule through the predictor stage, an extra accumulator pair at the same $\mathcal{O}(N^2)$ time and $\mathcal{O}(NBD)$ memory; we leave its derivation to future work.

B.4 Sum-of-Exponentials Kernel

The quadratic forward comes from the history sums of Eq. 10. Writing the power-law kernel as a Laplace transform, $t^{\alpha-1} = \Gamma(1-\alpha)^{-1} \int_0^\infty \lambda^{-\alpha} e^{-\lambda t}\, \mathrm{d}\lambda$, and discretising it on K fixed log-spaced nodes λ_k turns both weight sequences into geometric mixtures, $w_i \approx \sum_{k=1}^{K} d_k(\alpha)\, e^{-\lambda_k i \Delta t}$, so each history sum becomes K running accumulators updated once per step [36, 55]; multifactor approximations of rough volatility rest on the same representation [1]. We keep the two most recent weights exact, where the kernel is singular, and obtain the coefficients $d_k(\alpha)$ by a relative-error least-squares fit to the exact weights, solved with a differentiable QR factorisation; because the nodes are fixed, the coefficients are explicit differentiable functions of α, which stays learnable. We use $K = 16$ nodes on $[0.1/T,\ 5/\Delta t]$. The weights then match the exact ones to a relative error of 1.7×10^{-5} for $\alpha \in \{0.55, 0.6, 0.75, 0.9\}$ at $N = 4096$, and simulated paths match the exact solver to 3.5×10^{-8} at $N = 100$, about six orders of magnitude below the Monte Carlo floor. The autograd α-gradient through the coefficients agrees with finite differences to a relative error of 8×10^{-8}. On a single CPU thread the fitted exponent of forward time in N is 1.77 for the exact kernel and 1.00 for the SOE kernel (over $N = 4096$–32768). Up to $N = 2048$ the two kernels' times differ by at most about 30%, so the exact kernel remains competitive on our grids ($N = 128$ in the option study and 400 in the synthetic study of Appendix C.3). From $N = 4096$ the SOE forward is 1.8 to 8.5 times faster. Training the toy of Section 4.1 through the SOE forward, with autograd as the backward pass, recovers $\hat{\alpha} = 0.7513$, against 0.7516 with the exact kernel (truth 0.75). We have not implemented an adjoint for this kernel. Substituting the approximated weights and their α-derivatives into our backward recursion would keep $\mathcal{O}(NBD)$ memory at $\mathcal{O}(N^2)$ time; a backward that exploits the exponential recursion to run in $\mathcal{O}(NK)$ is future work. A blockwise FFT convolution would accelerate the same history sums but breaks the step recursion that our adjoint exploits.

B.5 Adjoint Gradients Near $\alpha = \frac{1}{2}$

Section 4.1 compares adjoint and autograd gradients at five values of α in $[0.63, 0.98]$. Because the scheme's variance constant grows as α approaches $\frac{1}{2}$ (Appendix B.2) and the option calibrations reach $\hat{\alpha} = 0.586$, we repeat the comparison at $\alpha \in \{0.55, 0.58, 0.60\}$ ($N = 100$, $B = 64$, $D = 2$, single precision). The forward trajectories agree exactly (checked at $\alpha = 0.55$; the losses are bit-identical at all three), and the relative gradient differences are at most 1.1×10^{-5} for u_α, 6×10^{-7} for X_0, 1.8×10^{-7} for the drift parameters and 3.5×10^{-7} for the diffusion parameters; the adjoint's u_α-gradient also matches a central finite difference to within 0.16%.

C COVID-19 Supplementary

C.1 Full Real-Data Results

Table 6 reports each cell's $h{=}7$ RMSE computed from both the median of the squared errors (the statistic of Table 1) and their arithmetic mean (the usual RMSE). The two agree closely on most cells, but on three Neural SDE cells the mean is far larger than the median: UK Alpha cases (1.99×10^4 vs 81.1), UK Alpha deaths (7.5×10^{14} vs 1.25), and US Winter cases (335 vs 66.5). These reflect two of the 192 Neural SDE calibrations, UK Alpha origin 165 seed 1 (both channels) and US Winter origin 130 seed 0 (cases), which produced anomalously large forecasts under the exponential observation map, consistent with the absence of Caputo damping of the accumulated noise at the daily step when $\chi \equiv 1$ (Section 4.2). The Neural FSDE produced no such instabilities across the same 192 calibrations on the same architecture and seeds. The boxplot of per-cell squared-error distributions in Figure 7 shows these as visible high-tail outliers.

Table 6: Full $h{=}7$ forecast RMSE on the four OWID slices, reported as median (arithmetic mean) over 48 squared-error observations per cell. The median is the statistic shown in Table 1; the parenthetical mean exposes calibrations that produced anomalously large forecasts.

Slice	Parametric FSDE	Neural FSDE	Neural SDE	Persistence
Cases per million per day				
United Kingdom (Alpha)	104.7 (149.4)	74.4 (155.8)	81.1 (1.99×10^4)	75.2 (123.4)
United Kingdom (Delta)	49.2 (135.1)	63.0 (126.9)	72.0 (183.6)	60.3 (140.7)
United States (Winter)	75.6 (84.3)	57.6 (79.9)	66.5 (334.9)	74.8 (76.1)
United States (Delta)	47.3 (64.6)	34.3 (53.4)	31.8 (130.9)	33.0 (39.5)
Deaths per million per day				
United Kingdom (Alpha)	0.75 (2.19)	1.02 (3.29)	1.25 (7.5×10^{14})	1.23 (2.19)
United Kingdom (Delta)	0.23 (0.23)	0.19 (0.24)	0.18 (0.23)	0.16 (0.20)
United States (Winter)	0.36 (0.54)	0.41 (0.74)	0.48 (0.68)	0.52 (0.67)
United States (Delta)	0.63 (0.80)	0.67 (0.91)	0.51 (0.73)	0.53 (0.75)

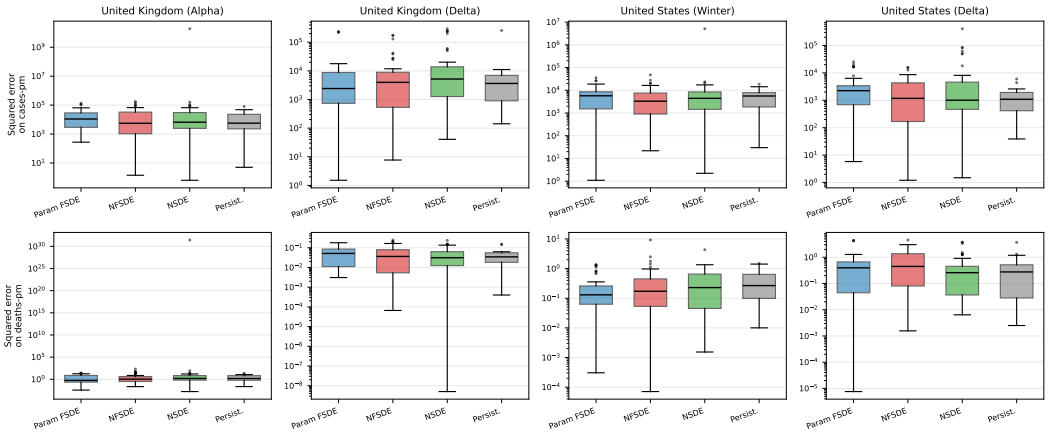


Figure 7: Per-cell distribution of squared $h{=}7$ forecast errors (log scale), 48 origin-seed observations per (slice, channel, method): top row cases per million per day, bottom row deaths per million per day. Boxes are 25th–75th percentile, whiskers extend to $1.5\times$IQR, dots are individual outliers. Visible Neural SDE outliers in UK Alpha (both channels) and US Winter cases reflect two of 192 calibrations producing anomalously large forecasts; the Neural FSDE has no comparable outliers despite sharing the same 3-D architecture, exponential observation map, and seed schedule, consistent with a stabilising effect from the $\chi < 1$ Caputo memory kernel. Axis labels: Param FSDE = Parametric FSDE, NFSDE = Neural FSDE, NSDE = Neural SDE, Persist. = Persistence.

We also ran seven baselines under the identical protocol: 60-day windows ending at each origin, 7-day inner validation for checkpointing, persistence-anchored evaluation and three seeds; for the three deterministic methods, ARFIMA, Chronos and TimesFM, the three seeds give identical forecasts. The baselines are a Neural CDE [39], a continuous-time sequence model; a Neural SVE [8], which

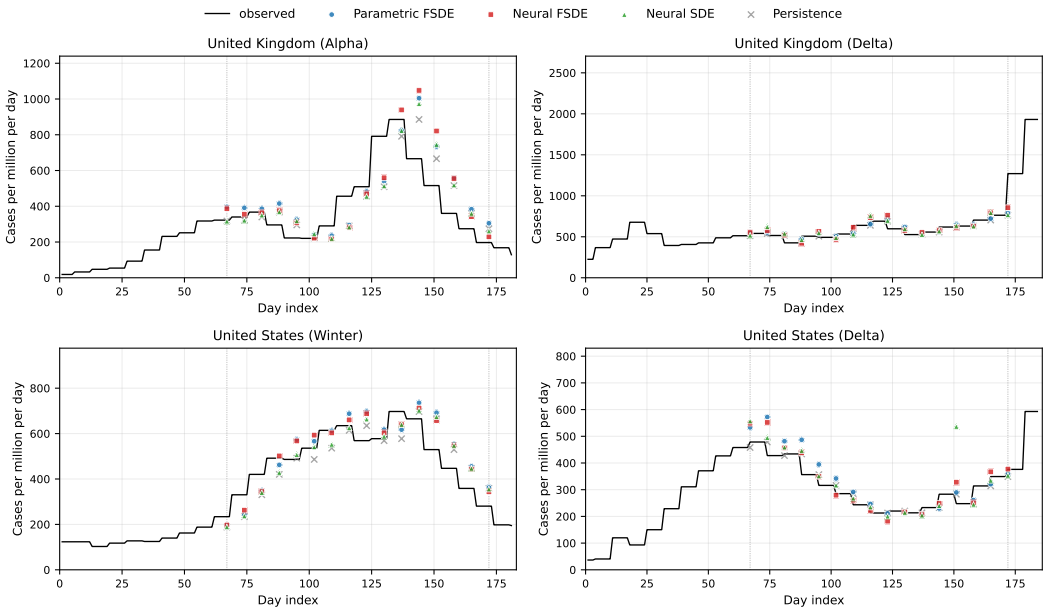


Figure 8: Real-world COVID-19 rolling-origin h=7 forecasts on cases per million per day across the four OWID slices. Black solid line shows observed daily cases; coloured markers show the median (across three random seeds) 7-day-ahead forecast at each of 16 weekly origins $t_0 \in \{60, 67, \ldots, 165\}$, plotted at horizontal coordinate $t_0 + 7$. Vertical dotted lines mark the first and last forecast endpoints. The Parametric FSDE retains Bonyah's 5-D compartmental structure; Neural FSDE and Neural SDE share a 3-D free-form latent with exponential observation map (only χ differs).

replaces the fixed power-law kernel by freely learned neural drift and diffusion kernels on a 3-D latent state, so that the shape of the memory kernel is learned from data (power-law kernels recover the Caputo model); ARFIMA [27, 35], the classical long-memory model, fitted per channel to the log series with its differencing order estimated by local Whittle [49]; an LSTM [33] and a causal Transformer [54]; and two pretrained zero-shot forecasters, Chronos [3] and TimesFM [15], conditioned on the same 60-day context. When the differencing order estimated on the log series is at least 0.7, which holds in 123 of the 128 fits, we estimate $\hat{d}$ on the first differences and use the order $1 + \hat{d}$. The LSTM (hidden width 64), the Transformer (width 32, two layers, two heads) and the Neural CDE (latent dimension 16, hidden width 32, two Euler substeps per day) are trained for 400 Adam steps with learning rates 10^{-2}, 3×10^{-3} and 3×10^{-3} and gradient clip 1; the Neural SVE's kernels are two-hidden-layer networks of width 32 over the lag, initialised nearly flat and trained at learning rate 10^{-3}, with everything else as for the Neural FSDE; the released code lists every setting. We report the pretrained models out of competition, because their public training corpora plausibly contain COVID-19 series. Table 7 gives the median RMSE of all eleven methods at $h = 7$.

Each cell of Table 1 summarises 48 (origin, seed) forecasts, but the three seeds of an origin share its data window and are not independent, so we test at the level of the 16 weekly origins. For a pair of methods we average the squared-error differential over the seeds of each origin; origins are seven days apart and $h = 7$, so consecutive forecasts do not overlap. The Diebold-Mariano test [17] with the small-sample correction of Harvey et al. [31] (lag-0 variance and effective horizon $h_{\text{eff}} = 1$) is then a paired t-test on these 16 differentials; a variant with a 6-lag heteroskedasticity- and autocorrelation-consistent (HAC) variance over the origins checks robustness to residual serial dependence. This variant uses Bartlett weights where the rectangular estimate is not positive and applies the small-sample factor at $h = 7$, which makes it conservative. Two distribution-free tests complement it: an origin-clustered paired bootstrap of the median-RMSE difference (the 16 origins are resampled with replacement, each carrying its three seeds; $10{,}000$ resamples) and an exact sign test on the 16 seed-averaged differentials.

Table 8 reports the four pairings that involve an FSDE variant: no difference is significant at the 5% level (minimum $p = 0.057$, UK Delta cases, favouring the Neural FSDE over the Neural SDE; minimum HAC $p = 0.056$). On UK Alpha, both differentials against the Neural SDE (cases and

Table 7: COVID-19 median 7-day-ahead RMSE (per million per day) for the four methods of Table 1 and seven additional baselines under the identical rolling-origin protocol (16 origins $\times$ 3 seeds per cell). Bold: best per column over all eleven methods. †Pretrained zero-shot models, reported out of competition.

	UK Alpha		UK Delta		US Winter		US Delta	
Method	cases	deaths	cases	deaths	cases	deaths	cases	deaths
Parametric FSDE	104.7	**0.751**	**49.2**	0.227	75.6	**0.360**	47.3	0.628
Neural FSDE	74.4	1.025	63.0	0.190	**57.6**	0.415	34.3	0.668
Neural SDE	81.1	1.251	72.0	0.177	66.5	0.478	31.8	0.508
Persistence	75.2	1.231	60.3	0.160	74.8	0.516	33.0	0.525
Neural CDE	83.5	1.568	87.7	0.314	100.1	0.525	72.8	1.566
Neural SVE	72.8	1.519	57.4	0.259	59.1	0.529	36.2	0.610
ARFIMA	**68.7**	1.514	53.5	0.180	58.4	0.508	26.8	0.436
LSTM	83.8	1.393	71.0	0.260	93.8	0.909	30.4	0.907
Transformer	87.0	1.884	72.0	0.210	73.2	0.556	38.4	0.805
Chronos†	80.2	1.254	62.1	**0.132**	71.6	0.549	**20.5**	0.460
TimesFM†	70.0	1.133	63.0	0.179	73.3	0.669	36.5	**0.421**

deaths) are dominated by the Neural SDE's anomalous forecasts at one origin (Section 4.2), which pins $|t|$ at one: a single outlier among n differentials gives $|t| \approx 1$ regardless of its size, so the test cannot register such failures. The distribution-free tests show no significant advantage for the Neural FSDE either: for these pairings the origin-clustered bootstrap interval of the median-RMSE difference contains zero in all 32 (pairing, cell) combinations, and the origin-level sign test reaches $p < 0.05$ in two of them, both favouring the comparator: on UK Delta deaths against Persistence and on US Winter deaths against the Parametric FSDE, the Neural FSDE is lower on 3 of 16 origins. The COVID differences are therefore directional, and the statistical weight of our comparisons rests on the option study (Appendix D.1).

Table 8: Diebold-Mariano tests with the Harvey-Leybourne-Newbold small-sample correction for the COVID-19 rolling-origin forecasts at $h = 7$. For each cell the loss differential is the per-origin, seed-averaged difference in squared forecast error ($n = 16$ weekly origins; origins are seven days apart, so consecutive forecasts do not overlap and the lag-0 variance estimator applies). Negative t favours the first-named method; p-values are two-sided from t_{15}. $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

		Neural FSDE vs. Neural SDE		Neural FSDE vs. Persistence		Param FSDE vs. Persistence		Neural FSDE vs. Param FSDE	
Slice	Channel	t	p	t	p	t	p	t	p
United Kingdom (Alpha)	Cases	-1.00	0.333	$+1.35$	0.197	$+1.50$	0.154	$+0.57$	0.579
	Deaths	-1.00	0.333	$+1.17$	0.260	-0.01	0.990	$+1.37$	0.190
United Kingdom (Delta)	Cases	-2.06	0.057	-0.56	0.583	-0.75	0.468	-0.47	0.647
	Deaths	$+0.48$	0.641	$+1.41$	0.179	$+1.07$	0.302	$+0.25$	0.805
United States (Winter)	Cases	-0.99	0.338	$+0.39$	0.702	$+0.91$	0.377	-1.44	0.170
	Deaths	$+0.55$	0.593	$+0.37$	0.718	-1.03	0.318	$+1.58$	0.136
United States (Delta)	Cases	-1.61	0.128	$+1.66$	0.118	$+2.02$	0.062	-1.42	0.175
	Deaths	$+1.87$	0.080	$+1.67$	0.116	$+0.97$	0.346	$+1.21$	0.244

Table 9 gives the significance of each additional baseline against the Neural FSDE and the Parametric FSDE under the three tests. No method wins uniformly. Among the methods fitted afresh on each window, the two FSDE variants are best in the most cells (four of eight), ARFIMA in three and Persistence in one; with the pretrained models included, Chronos is best in two cells and TimesFM in one. No baseline is more accurate than the Neural FSDE in any cell under both the Diebold-Mariano test and the clustered bootstrap. The only test at the 5% level favouring a baseline over the Neural FSDE, TimesFM on US Delta deaths (DM $p = 0.030$), has a bootstrap interval that contains zero and does not survive a Holm correction across the eight cells (adjusted $p = 0.24$); every other significant difference favours the Neural FSDE. Against the Parametric FSDE, ARFIMA is more accurate on UK Alpha cases under the Diebold-Mariano test (DM $p = 0.003$, Holm-adjusted 0.026; lower on 14 of 16 origins), though its clustered bootstrap interval, $[-55.7, +4.2]$, contains zero, so on the cases channel this classical long-memory model competes with the mechanistic drift. The Neural

CDE is the least stable method: 5 of its 192 fits produce an anomalous 7-day forecast, defined as a squared error above 100 times the largest persistence error in the cell, and 24 do so at the 14-day horizon, which we also evaluated. The Neural SDE has 2 such fits at either horizon, and the two FSDE variants, the Neural SVE and the remaining baselines have none. The Neural SVE's learned drift kernel decays mildly, far more gently than a power-law singularity: it is monotone in 81% of fits, and the median ratio of its value at the longest lag on the simulation grid, 72 days, to that at lag 0 is 0.87.

Table 9: Significance of the additional COVID-19 baselines against the Neural FSDE and the Parametric FSDE over the eight (slice, channel) cells at $h = 7$. *Lower*: cells where the baseline's median RMSE is lower. The other columns count cells significant at 5% as *favouring the baseline* / *favouring the FSDE model*: Diebold-Mariano test with the Harvey-Leybourne-Newbold (HLN) small-sample correction on the 16 seed-averaged origin differentials (DM), origin-clustered paired bootstrap interval of the median-RMSE difference excluding zero (CI), and exact sign test over the 16 origins (sign). †Out of competition.

	vs. Neural FSDE				vs. Parametric FSDE			
Baseline	lower	DM	CI	sign	lower	DM	CI	sign
Neural CDE	0/8	0/0	0/4	0/2	1/8	0/0	0/3	0/2
Neural SVE	3/8	0/1	0/0	0/1	4/8	0/1	0/0	0/1
ARFIMA	5/8	0/0	0/0	0/0	5/8	1/0	0/1	1/0
LSTM	1/8	0/1	0/2	0/1	2/8	0/2	0/2	0/1
Transformer	0/8	0/0	0/1	0/0	4/8	0/2	0/3	0/3
Chronos†	4/8	0/0	0/0	0/0	5/8	0/0	0/1	0/0
TimesFM†	3/8	1/0	0/0	0/0	5/8	0/0	0/0	0/0

C.2 Recovered Bonyah Parameters

Table 10 records the ten recovered rate scalars of Bonyah's compartmental drift (Eq. (7) of Bonyah et al.) and the fractional order χ from the Parametric FSDE configuration on each OWID slice, summarised across the 48 rolling-origin calibrations as per-slice medians, alongside the values tabulated by Bonyah et al. (their Table 2), which are the ground truth of our synthetic study (Appendix C.3). Recruitment Π and transmission rate β in our fit are in per-million-population units and so differ from Bonyah's absolute-population values by the population-size factor. The noise intensities $\Lambda_{j,k}$ of Bonyah's diffusion (Eq. (9) of Bonyah et al.) are not tabulated because the 60-day rolling windows are too short for the structured diffusion to move appreciably from initialisation across the 192 calibrations.

Table 10: Median recovered rate scalars of Bonyah's compartmental drift (Eq. (7) of Bonyah et al.) and fractional order χ from the Parametric FSDE configuration on each OWID slice (median across 48 rolling-origin calibrations per slice), alongside the values tabulated by Bonyah et al. (their Table 2; column Bonyah T2), the ground truth of our synthetic study. Symbols follow Bonyah et al. and are unrelated to our drift μ, diffusion σ and network parameters θ. Π and β in our fit are in per-million-population units. The fractional order χ falls below 1 on every slice and on every one of the 192 individual calibrations (full per-calibration range $[0.58, 0.81]$).

Symbol	Description	UK (Alpha)	UK (Delta)	US (Winter)	US (Delta)	Bonyah T2
Bonyah system-(7) rate scalars						
Π	recruitment rate	948.8	249.2	1.08×10^{3}	701.5	0.6
β	transmission rate	9.99×10^{-8}	9.44×10^{-8}	9.90×10^{-8}	9.37×10^{-8}	0.6
μ	mortality rate	6.90×10^{-4}	2.15×10^{-4}	8.38×10^{-4}	6.32×10^{-4}	0.1
θ_1	T-class infectiousness	0.543	0.503	0.491	0.450	0.8
θ_2	A-class infectiousness	0.472	0.454	0.474	0.440	0.8
σ	E exit rate	0.0909	0.0896	0.0910	0.0849	0.6
α_T	treatment-recovery rate	0.105	0.108	0.105	0.107	0.4
ν_1	I $\to$ T progression	0.0542	0.0571	0.0538	0.0574	0.04
ν_2	I $\to$ R recovery	4.50×10^{-3}	4.68×10^{-3}	4.65×10^{-3}	4.93×10^{-3}	5.00×10^{-3}
ρ	fraction symptomatic	0.552	0.515	0.554	0.516	0.6
χ	fractional order	0.706	0.679	0.700	0.673	—

C.3 Synthetic χ-Recovery Trajectories

We fit four configurations, all on Bonyah's 5-D state, to paths simulated from the Bonyah model: the Parametric FSDE of Section 4.2; Hybrid, which keeps Bonyah's drift and replaces the Atangana-Seda diffusion by a neural diffusion; Full-neural, a Neural FSDE with neural drift and diffusion; and a 5-D Neural SDE, the Full-neural model with $\chi \equiv 1$, which is not the 3-D latent Neural SDE of Section 4.2. We call the first two structured-drift and the last two flexible-drift configurations. Table 11 reports the recovered $\hat{\chi}$ and sliced-1-Wasserstein loss for each configuration across the three ground-truth values $\chi^* \in \{0.6, 0.75, 0.9\}$. Figure 9 shows the per-step training loss (top row) and the learnable χ trajectory (bottom row) over the 800 Adam steps, for each configuration and each χ^*. Figure 10 shows the path-mean and $\pm 1\sigma$ envelopes for the five compartments at $\chi^* = 0.75$. The Parametric FSDE fits both the path means and the $\pm 1\sigma$ envelopes closely, and the Hybrid fits the path means with a visibly narrower envelope; the two flexible-drift configurations miss the early transient and match the path means only after roughly the first two weeks, at a much higher loss (Table 11). Identification is judged by whether the recovered $\hat{\chi}$ matches χ^*.

The two structured-drift configurations recover χ^* to within $|\hat{\chi} - \chi^*| \leq 0.01$ on every χ^*. The Parametric FSDE also recovers eight of the ten Bonyah rate scalars to within a few percent of the values of Bonyah et al. (their Table 2); the bilinear infectiousness modifiers $\theta_{1,2}$ are the loosest, partially compensated by β. The Full-neural configuration shows the failure of identification noted in Section 4.2: with Bonyah's parametric drift replaced by a neural one, $\hat{\chi}$ moves towards 1 regardless of χ^*, while the 5-D Neural SDE has $\chi \equiv 1$ by construction. An unrestrained MLP drift can fit a Markov approximation $(\mu'_\theta, \sigma'_\phi, \chi' \to 1)$ matching the truth's path mean at a higher residual loss. The structured drift class breaks the (μ, σ, χ) non-uniqueness and lets χ be identified, which is why we read the recovered $\hat{\chi}$ off the Parametric FSDE configuration on real OWID data (Appendix C.2).

Table 11: Synthetic recovery: ground-truth χ^* vs. learned $\hat{\chi}$ across the four configurations on paths simulated from the Bonyah model. The two structured-drift configurations recover each χ^* closely; Full-neural fails to identify χ, and the 5-D Neural SDE has $\chi \equiv 1$ by construction. Full-neural and Neural SDE here act on the 5-D Bonyah state and are not the 3-D latent models of Section 4.2.

Configuration	χ^*	$\hat{\chi}$	$\lvert\hat{\chi} - \chi^*\rvert$	Sliced 1-W loss
Parametric FSDE	0.60	0.607	0.007	0.0561
Parametric FSDE	0.75	0.751	0.001	0.0438
Parametric FSDE	0.90	0.896	0.004	0.0357
Hybrid	0.60	0.605	0.005	0.2412
Hybrid	0.75	0.756	0.006	0.1910
Hybrid	0.90	0.901	0.001	0.1451
Full-neural	0.60	0.990	0.390	1.2227
Full-neural	0.75	0.990	0.240	1.0786
Full-neural	0.90	0.990	0.090	0.9319
Neural SDE	0.60	1.000	0.400	1.2574
Neural SDE	0.75	1.000	0.250	1.1153
Neural SDE	0.90	1.000	0.100	0.9562

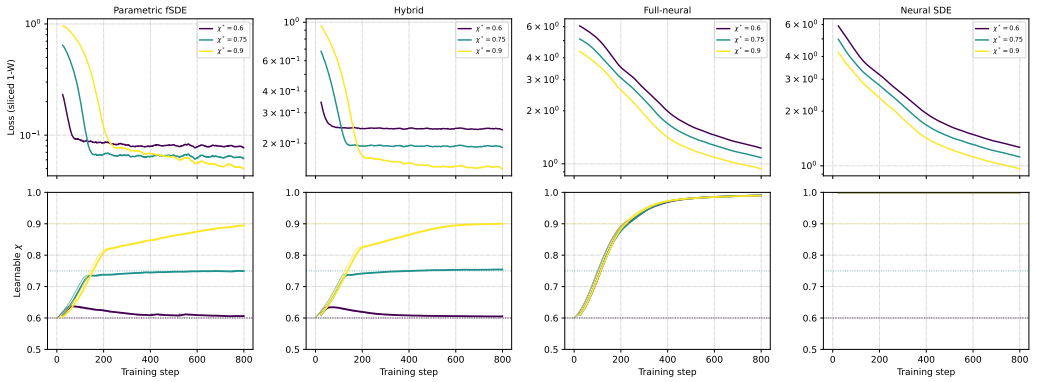


Figure 9: Synthetic recovery on paths simulated from the Bonyah model: training loss (top) and learnable χ (bottom) over 800 Adam steps. Columns left to right: Parametric FSDE, Hybrid, Full-neural, 5-D Neural SDE ($\chi \equiv 1$). Three colours per panel = three ground-truth values $\chi^* \in \{0.6, 0.75, 0.9\}$; dashed lines mark each target. In the two structured-drift columns the learnable χ converges to its target; in the Full-neural column it rises towards 1 for every χ^*, and in the 5-D Neural SDE column it is fixed at 1.

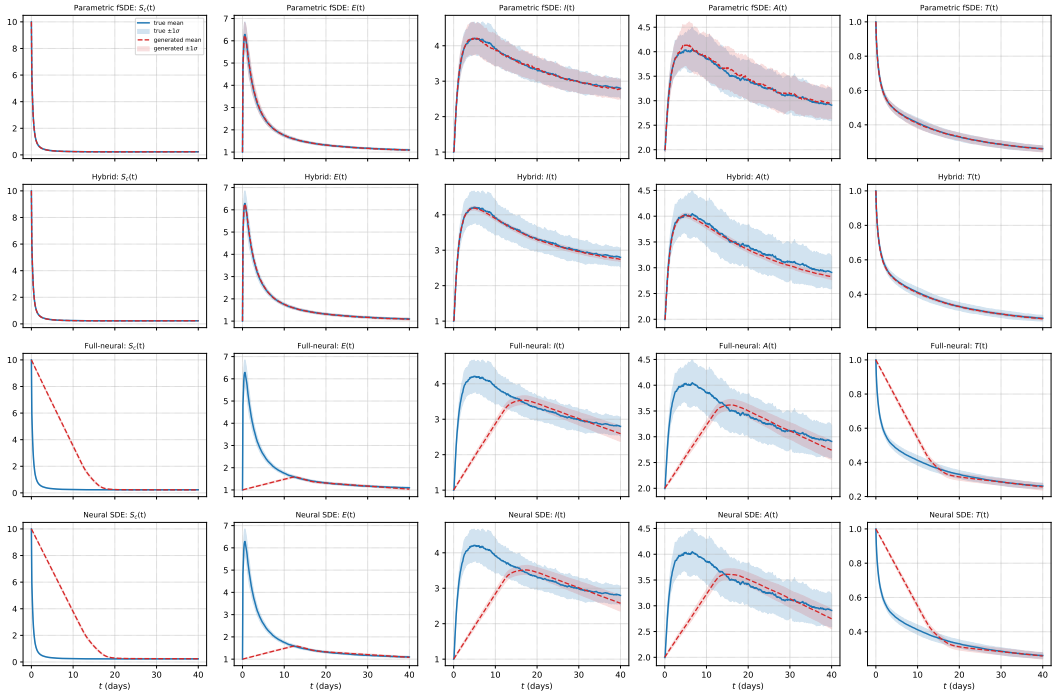


Figure 10: Synthetic recovery, $\chi^* = 0.75$: path-mean and $\pm 1\sigma$ envelopes for the five compartments (S_c, E, I, A, T). One row per configuration. The Parametric FSDE fits both the path means and the envelopes closely, and the Hybrid fits the path means with a visibly narrower envelope; the two flexible-drift configurations miss the early transient and match the path means only after roughly the first two weeks, at a much higher loss (Table 11).

C.4 Training Details

Network architecture: neural drift and diffusion heads (used by Neural FSDE, Neural SDE, and the synthetic Hybrid configuration) are 3-layer time-invariant MLPs with hidden width 64 and $\tanh$ activations. Where learnable, the fractional order is parametrised as $\chi = \frac{1}{2} + \frac{1}{2}\,\mathrm{sigmoid}(u_\chi)$ and initialised at $\chi = 0.6$ for synthetic verification, $\chi = 0.7$ for real-data forecasting. Final-layer weights of σ_ϕ (Neural FSDE / Neural SDE / Hybrid) are zero-initialised so a fresh model starts deterministic; in the synthetic study those of μ_θ (Full-neural / 5-D Neural SDE) are zero-initialised too, so it also

starts driftless. *Synthetic optimisation*: Adam with 800 steps, training batch $B_{\text{train}} = 64$ truth paths per step (resampled from a truth set of $B_{\text{truth}} = 128$), $\text{lr}_{\text{net}} = 10^{-3}$ and $\text{lr}_{\chi} = 10^{-2}$, gradient norm clip 0.5, and a `NaN`-step rollback to the previous best state. *Real-data optimisation*: per-window Adam with 400 steps and $B = 32$ Monte Carlo paths per step on the $N = 60$-day inner-train block (last 7 days reserved as inner-validation for best-checkpoint selection), $\text{lr}_{\text{net}} = 10^{-3}$, $\text{lr}_{\chi} = 5\times10^{-3}$, $\text{lr}_{\text{IC}} = 5\times10^{-2}$ for the free initial-condition magnitudes, weight decay 10^{-3}, gradient norm clip 1.0, and the diffusion sub-network frozen for the first 200 steps. Compute: a single NVIDIA H100 GPU for both the synthetic recovery (four configurations $\times$ three ground-truth χ^*) and the rolling-origin real-data sweep (3 calibratable configurations $\times$ 4 slices $\times$ 16 origins $\times$ 3 seeds $= 576$ calibrations).

C.5 Window-Length Sensitivity

Section 4.2 fixes the training window at N=60 days. To check this choice, we re-ran the rolling-origin protocol for the three calibrated methods on US Winter (one of the two slices on which the Neural FSDE is best on cases in Table 1) at $N \in \{60, 90, 120\}$ days. The origin set is restricted to $\{123, 130, 137, 144, 151, 158, 165\}$, the seven origins ≥ 119 for which N=120 has a valid window start, with three seeds each, giving 189 calibrations in total.

Table 12 and Figure 11 report the cell-wise median-RMSE. All three methods are best at N=60 on the cases channel, with errors growing monotonically as the window lengthens: the Neural FSDE's cases RMSE grows $65.8 \to 74.3 \to 88.5$ across N, the Parametric FSDE's $82.8 \to 102.6 \to 107.8$, and the memoryless Neural SDE's $84.2 \to 89.2 \to 90.8$. On the deaths channel the Parametric FSDE and the Neural SDE change little between N=60 and N=90, while the Neural FSDE already worsens at N=90; at N=120 both FSDE variants are less accurate than the Neural SDE and Persistence. Persistence, which does not train and is therefore unaffected by N, is a flat 82.5 / 0.55 reference across windows. The pattern is consistent with an underlying epidemic state evolving on a multi-week timescale: a longer training window dilutes the signal from the most recent 14–21 days that drive the next h=7 forecast.

Table 12: Window-length sensitivity on US Winter at $h = 7$: median-RMSE for each method on the seven origins ≥ 119 that are valid for $N \in \{60, 90, 120\}$. Cases and deaths are per million per day, matching Table 1. The $N = 60$ column covers the same slice as Table 1; the seven-origin restriction explains the small numerical shift from its 16-origin median.

	Cases (per million per day)			Deaths (per million per day)		
Method	N=60	N=90	N=120	N=60	N=90	N=120
Param FSDE	82.8	102.6	107.8	0.389	0.416	0.635
Neural FSDE	65.8	74.3	88.5	0.390	0.547	0.614
Neural SDE	84.2	89.2	90.8	0.487	0.489	0.550
Persistence	82.5	82.5	82.5	0.550	0.550	0.550

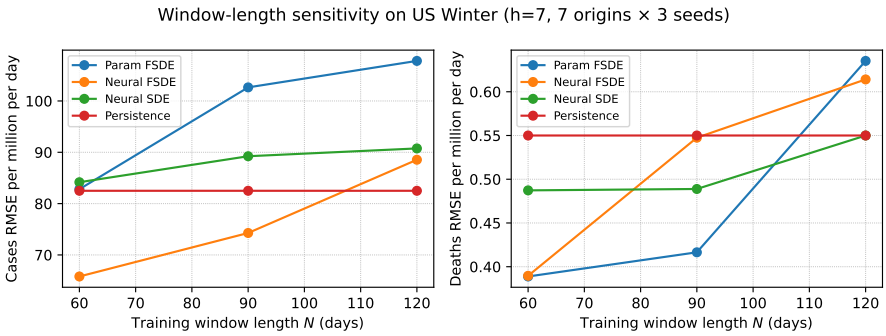


Figure 11: Window-length sensitivity on US Winter at h=7, seven origins $\times$ three seeds. Left panel: cases-channel median-RMSE (per million per day) vs. training window N. Right panel: deaths channel (per million per day).

D Option Pricing Supplementary

D.1 Full Results

Tables 13 and 14 report the full per-quadrant breakdowns underlying the aggregate "all"-quadrant numbers in Table 2. Quadrants are short-near-money (sn: $\tau \leq 60$d, $|k| \leq 0.05$), short-far-money (sf: $\tau \leq 60$d, $|k| > 0.05$), long-near-money (ln: $\tau > 60$d, $|k| \leq 0.05$), and long-far-money (lf: $\tau > 60$d, $|k| > 0.05$).

Table 13: SPXW per-quadrant IV-RMSE (bps, median across 483 days) under the four evaluation schedules. Neural SDE at its best diffusion bound $c = 12$ (Appendix D.4). Bold = column-best within each schedule block.

Schedule	Model	all	sn	sf	ln	lf
(a) In-sample	rough Bergomi	585	**171**	974	157	326
	rough Heston	344	221	515	**115**	**120**
	PDV	450	306	582	413	334
	Neural SDE	348	207	507	188	240
	Neural FSDE	**216**	172	**280**	138	145
(b) Day-ahead OOS	rough Bergomi	727	**204**	1183	329	506
	rough Heston	347	227	511	**116**	**122**
	PDV	524	318	693	436	342
	Neural SDE	382	208	550	209	254
	Neural FSDE	**269**	225	**342**	169	193
(c) Expiry-stratified	rough Bergomi	407	**153**	698	247	308
	rough Heston	232	212	342	**98**	**107**
	PDV	388	293	379	479	402
	Neural SDE	285	198	319	367	263
	Neural FSDE	**212**	157	**228**	263	213
(d) Liquidity-stratified	rough Bergomi	537	**162**	880	139	319
	rough Heston	296	216	420	114	**118**
	PDV	391	240	539	300	251
	Neural SDE	328	206	453	163	251
	Neural FSDE	**212**	176	**271**	**107**	144

Table 15 reports paired-bootstrap 95% confidence intervals (10,000 resamples) and exact-binomial sign-test p-values for four pairings on each of the eight option-pricing settings of Table 2. The bootstrap resamples calibration days with replacement, keeping each day's pair of errors together (RNG seed 42); its statistic is the difference in median IV-RMSE in basis points, and CIs are read off the 2.5^{th}/97.5^{th} empirical quantiles of the bootstrap distribution. The exact-binomial sign test reports the two-sided p-value for the count of days on which the Neural FSDE achieved a strictly lower per-day IV-RMSE than the comparator, under the null $p = 0.5$.

All 32 confidence intervals at 95% are strictly below zero in favour of the Neural FSDE, and all 32 sign-test p-values fall below 10^{-17}. The smallest sign-test count is $336/483$, on SPXW (c) against rough Heston ($p = 4.4 \times 10^{-18}$), and three of the 32 pairings are unanimous ($483/483$ or $482/482$ days, all against rough Bergomi), where the two-sided binomial p-value is at its floor $2\,(0.5)^n$. Against PDV the Neural FSDE is lower on 441–479 days per setting. Against the architecture-matched Neural SDE at its best bound it is lower on 435–480 days per setting, and the upper end of the 95% CI is at least 60 bps below zero on every setting; at the common bound $c = 3$ it is at least 160 bps below zero (Appendix D.4).

Figure 12 shows the per-day distribution of IV-RMSE (boxplots) by model and quadrant on SPXW (a), and Figure 13 shows representative smile fits on a single SPXW day.

D.2 Calibration and Evaluation Details

Each dataset row is preprocessed with implied volatility, log-moneyness $k = \log(K/F)$, forward F, and a per-quote risk-free rate, so the loader does not need to invert Black-Scholes at load time.

Table 14: SPY per-quadrant IV-RMSE (bps, median across 483 days) under the four evaluation schedules. Each model is calibrated directly on SPY quotes from a cold start; implied volatilities invert the Bjerksund-Stensland 2002 approximation at the least-squares Monte Carlo (LSM) American price (Appendix D.2). Neural SDE at its best diffusion bound $c = 12$ (Appendix D.4). Bold = column-best within each schedule block.

Schedule	Model	all	sn	sf	ln	lf
(a) In-sample	rough Bergomi	444	183	787	195	305
	rough Heston	211	209	307	**155**	**121**
	PDV	263	216	340	255	183
	Neural SDE	244	162	353	225	203
	Neural FSDE	**165**	**127**	**228**	166	125
(b) Day-ahead OOS	rough Bergomi	448	202	788	197	310
	rough Heston	223	222	320	**159**	**126**
	PDV	283	229	368	259	184
	Neural SDE	260	181	362	236	208
	Neural FSDE	**182**	**149**	**248**	176	134
(c) Expiry-stratified	rough Bergomi	384	186	679	234	287
	rough Heston	207	208	303	**166**	**125**
	PDV	261	205	341	257	190
	Neural SDE	244	162	321	274	194
	Neural FSDE	**175**	**124**	**222**	200	138
(d) Liquidity-stratified	rough Bergomi	508	176	906	175	310
	rough Heston	206	207	**313**	155	120
	PDV	292	170	492	171	173
	Neural SDE	278	138	493	168	142
	Neural FSDE	**187**	**115**	320	**123**	**101**

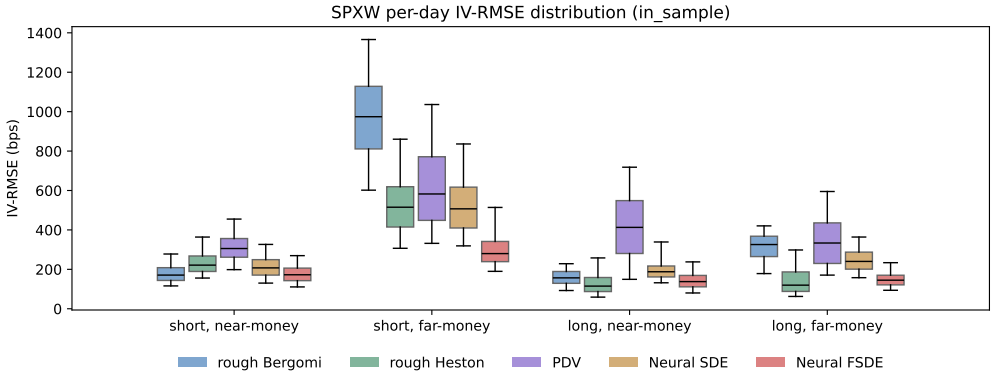


Figure 12: SPXW: per-day IV-RMSE distribution (in-sample) by model and quadrant. Boxes span the inter-quartile range; whiskers the 5th–95th percentile; outliers suppressed. The wing quadrant (*short, far-money*) is the hardest to fit for every model.

Calibration optimisation: Adam with 60 steps, $B = 2048$ Monte Carlo paths per step on a uniform $N = 128$ time grid over $[0, 1\text{y}]$, $\text{lr}_{\text{net}} = 5\times10^{-3}$ for the network parameters and $\text{lr}_{\alpha} = 2\times10^{-2}$ for u_{α}, gradient norm clip 1.0. The variance paths of rough Heston, the Neural SDE and the Neural FSDE are differentiated with the discrete adjoint of Section 3.3. The α-gradient of that adjoint treats the diffusion as independent of α, whereas the diffusion intensity $\eta\sqrt{2H}\,\Gamma(\alpha)$ of rough Bergomi depends on α, so rough Bergomi is differentiated by automatic differentiation, as are PDV and, for every model, the spot path and the pricer. Brownian draws are seeded from a hash of (date, model) for reproducibility, with a common-random-number rewind every step so the loss surface in model parameters is smooth. The held-out expiries of schedule (c) are drawn from a hash of the date alone, so every model is scored on the same quotes. Every final evaluation uses 32,768 paths (see below),

Table 15: Paired comparisons for Table 2: paired-bootstrap 95% CI (10,000 resamples) on the median-IV-RMSE difference Δ in basis points, and, out of the N common days per cell, the number on which the Neural FSDE has the lower IV-RMSE, with its exact binomial sign test. The Neural SDE rows use its best diffusion bound, $c = 12$ (Appendix D.4); they and the PDV rows use their own random-number streams for the bootstrap. Negative Δ favours Neural FSDE. Sign-test stars: $^{*}\, p < 0.05$; $^{**}\, p < 0.01$; $^{***}\, p < 0.001$.

Cell	N	Pairing	Δ (bps)	95% CI	Sign	p-value
SPXW (a)	483	Neural FSDE vs. rough Heston	-128.4	$[-139.2,\ -120.4]$	$479/483^{***}$	$<10^{-100}$
		Neural FSDE vs. Neural SDE	-132.4	$[-139.8,\ -124.7]$	$476/483^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-369.2	$[-384.8,\ -354.9]$	$483/483^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-234.3	$[-249.3,\ -212.0]$	$479/483^{***}$	$<10^{-100}$
SPXW (b)	482	Neural FSDE vs. rough Heston	-78.0	$[-89.8,\ -65.3]$	$373/482^{***}$	7.8×10^{-35}
		Neural FSDE vs. Neural SDE	-113.2	$[-125.8,\ -100.3]$	$435/482^{***}$	8.7×10^{-80}
		Neural FSDE vs. rough Bergomi	-458.2	$[-472.8,\ -441.0]$	$481/482^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-254.8	$[-279.1,\ -237.3]$	$448/482^{***}$	3.0×10^{-93}
SPXW (c)	483	Neural FSDE vs. rough Heston	-20.0	$[-32.0,\ -7.5]$	$336/483^{***}$	4.4×10^{-18}
		Neural FSDE vs. Neural SDE	-73.0	$[-79.7,\ -61.0]$	$447/483^{***}$	2.6×10^{-91}
		Neural FSDE vs. rough Bergomi	-195.0	$[-218.5,\ -175.1]$	$461/483^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-175.5	$[-193.7,\ -159.4]$	$465/483^{***}$	$<10^{-100}$
SPXW (d)	483	Neural FSDE vs. rough Heston	-84.5	$[-92.2,\ -75.8]$	$461/483^{***}$	$<10^{-100}$
		Neural FSDE vs. Neural SDE	-116.1	$[-123.5,\ -107.0]$	$479/483^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-325.5	$[-337.9,\ -314.4]$	$483/483^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-179.1	$[-191.8,\ -163.4]$	$472/483^{***}$	$<10^{-100}$
SPY (a)	482	Neural FSDE vs. rough Heston	-44.5	$[-52.1,\ -37.1]$	$454/482^{***}$	3.4×10^{-100}
		Neural FSDE vs. Neural SDE	-78.6	$[-84.3,\ -74.4]$	$480/482^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-278.1	$[-296.1,\ -263.0]$	$482/482^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-97.7	$[-107.3,\ -90.5]$	$471/482^{***}$	$<10^{-100}$
SPY (b)	482	Neural FSDE vs. rough Heston	-41.2	$[-49.2,\ -34.0]$	$416/482^{***}$	4.0×10^{-63}
		Neural FSDE vs. Neural SDE	-78.4	$[-83.3,\ -72.0]$	$476/482^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-266.2	$[-286.3,\ -252.0]$	$477/482^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-101.6	$[-114.9,\ -90.3]$	$452/482^{***}$	8.0×10^{-98}
SPY (c)	483	Neural FSDE vs. rough Heston	-32.3	$[-41.4,\ -26.2]$	$372/483^{***}$	5.8×10^{-34}
		Neural FSDE vs. Neural SDE	-69.1	$[-77.8,\ -64.0]$	$463/483^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-209.7	$[-227.3,\ -193.8]$	$478/483^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-86.6	$[-99.2,\ -77.1]$	$441/483^{***}$	5.3×10^{-85}
SPY (d)	483	Neural FSDE vs. rough Heston	-19.0	$[-27.6,\ -12.6]$	$354/483^{***}$	2.8×10^{-25}
		Neural FSDE vs. Neural SDE	-90.8	$[-100.4,\ -82.5]$	$476/483^{***}$	$<10^{-100}$
		Neural FSDE vs. rough Bergomi	-321.1	$[-333.5,\ -301.6]$	$482/483^{***}$	$<10^{-100}$
		Neural FSDE vs. PDV	-105.5	$[-114.8,\ -94.7]$	$461/483^{***}$	$<10^{-100}$

and the SPXW (b) re-pricing of day-$(t{+}1)$ quotes reuses the evaluation seed of the day-t fit. Compute: all option-pricing experiments ran on a single NVIDIA H100 GPU.

PDV [30] models volatility as an explicit function of the path, $\sigma_t = \beta_0 + \beta_1 R_{1,t} + \beta_2 \sqrt{R_{2,t}}$, where R_1 and R_2 mix a fast and a slow exponentially weighted average of past returns and of past squared returns. We use the four-factor Markovian (exponential-kernel) form, whose state updates in closed form, and calibrate its nine scalars (β_0, β_1, β_2, two mixing weights and four decay rates) per day to option quotes like the other models, whereas Guyon and Lekeufack estimate stationary parameters on historical returns. The initial state of the averages depends on the decay rates, so it is recomputed inside every gradient step from trailing daily returns: S&P 500 index closes before the sample (from January 2021) and, within the sample, the spot implied by each day's forward curve.

SPY options are American, so the pricer must decide on each simulated path when to exercise. We use the least-squares Monte Carlo (LSM) method of Longstaff and Schwartz [43], adapted to non-Markovian variance and to gradient-based calibration. The same pricer, basis and settings serve every model class.

Each model simulates the forward-moneyness process $X_t = S_t/F_{0,t}$, a martingale with $X_0 = 1$, together with its variance V_t, on the grid $t_n = n\Delta t$ with $\Delta t = 1/128$ year. The forward curve comes from the day's quotes: a least-squares fit of $\log F_{0,\tau} = \log S_0 + (r-q)\tau$ to the quoted forwards gives the spot S_0 and one carry rate $r - q$ that absorbs dividends and borrow costs, and the spot paths are $S_{t_n} = S_0\, e^{(r-q)t_n} X_{t_n}$. Cash flows are discounted at the day's mean risk-free rate r. Each

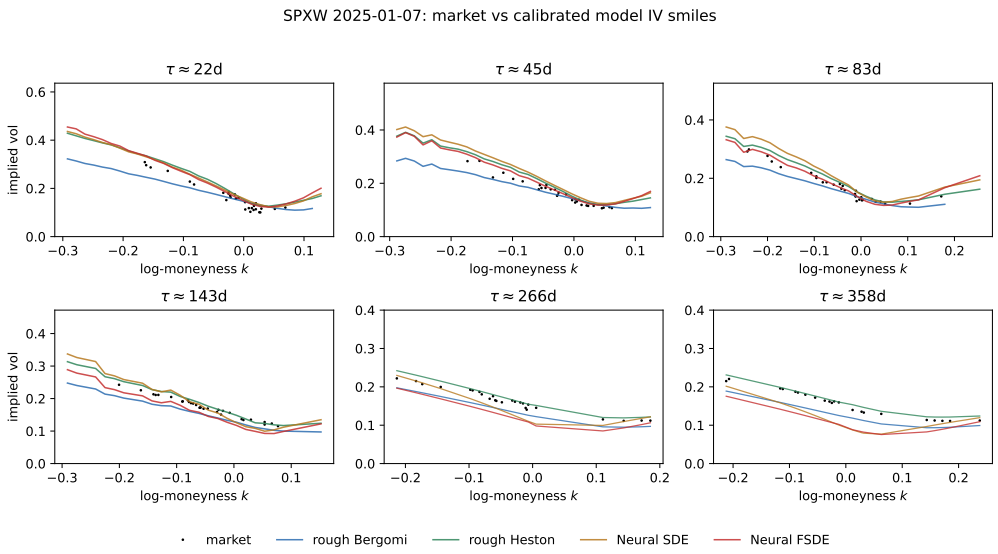


Figure 13: Calibrated IV smile vs market on a representative SPXW day (2025-01-07), one panel per maturity. Black dots: market quotes. Coloured curves: Neural FSDE / rough Bergomi / rough Heston / Neural SDE (PDV not shown).

option's maturity is rounded to the nearest grid point n_τ, and exercise is allowed at every grid point from t_1 to t_{n_τ}, a Bermudan approximation, in which exercise is allowed only on a discrete set of dates, here about 2.9 days apart.

For each option, every path's discounted cash flow C starts at its payoff at t_{n_τ}. Stepping back from $n = n_\tau - 1$ to $n = 1$, the continuation value is estimated by a ridge-regularised least-squares regression (ridge 10^{-3}) of C on all monomials of total degree at most two in the five features $(S_{t_n}, V_{t_n}, V_{t_{n-1}}, V_{t_{n-2}}, V_{t_{n-3}})$, each divided by its mean over paths (21 basis functions). A path exercises at t_n, and its cash flow becomes the discounted payoff, when the option is in the money and that payoff exceeds the regression estimate; the regression itself uses all paths, not only those in the money. Under a Markovian model the state (S_t, V_t) would suffice. The three lagged variances summarise the path history on which a non-Markovian model's continuation value depends, and every class uses the same basis, so no class gains from a richer regression. The price is the mean discounted cash flow over paths, with regression and valuation on the same paths (the in-sample estimator, without a dual upper bound).

Exercise decisions are discontinuous in the model parameters, so we differentiate with the exercise policy held fixed [26]. Each Adam step (i) simulates $B = 2048$ paths from a fixed seed without gradients, runs the backward induction, and records every path's exercise date for every option; then (ii) re-simulates the same paths with gradients and prices each option as the mean discounted payoff at the recorded dates. The loss of Section 4.3 is back-propagated through the simulator. Holding the policy fixed drops the dependence of the exercise boundary on the parameters. Because the optimal boundary maximises the price, that dependence has no first-order effect at the optimum, and approximately none at its LSM estimate; this is the usual justification of pathwise sensitivities for American options.

After calibration the price is recomputed on $B = 32{,}768$ fresh paths with a new regression and converted to an implied volatility by Newton inversion of the Bjerksund-Stensland approximation [9], with the fitted S_0 and carry $r - q$. The market implied volatilities follow the same convention. On 50 SPY days, changing the polynomial degree or the number of lagged variances moves the all-quadrant IV-RMSE by at most 14 bps, while the number of paths matters more; we calibrate with $B = 2048$ paths to bound the time and memory of the differentiable pricing and evaluate with 32,768, a choice examined next.

Every IV-RMSE in this paper is computed from Monte Carlo prices. A quote far from the money has a small vega, so a small error in its price becomes a large error in its implied volatility, and the Monte Carlo error inflates the reported IV-RMSE by an amount that decreases with the number of

evaluation paths B and differs between models: it is larger for models whose variance paths have heavier tails. To choose B, we re-scored stored schedule-(a) calibrations at 2^{13} to 2^{17} paths, each model with its own evaluation seed and without re-calibrating. The models are the five of Table 2 and the two anchor-free variants of Appendix D.5, whose drift omits the mean reversion towards $\log \xi_0$, and the days are every eighth trading day (61 SPXW days; 60 SPY days, one dropped because the Neural FSDE's calibration failed on it). Table 16 reports the medians.

Table 16: Median IV-RMSE (bps) of the stored schedule-(a) calibrations re-scored with B evaluation paths, on every eighth trading day (61 SPXW days, 60 SPY days), and the Neural FSDE's median minus those of rough Heston and of the Neural SDE (computed before rounding). The paper uses $B = 2^{15} = 32{,}768$.

	SPXW (a), B				SPY (a), B			
Model	2^{13}	2^{15}	2^{16}	2^{17}	2^{13}	2^{15}	2^{16}	2^{17}
rough Bergomi	658	587	538	526	494	429	425	405
rough Heston	386	343	322	312	231	213	212	208
PDV	509	463	433	415	289	261	260	257
Neural SDE, anchor-free	443	363	342	330	274	257	258	267
Neural FSDE, anchor-free	265	234	221	220	201	201	198	198
Neural SDE	393	341	331	322	255	243	242	246
Neural FSDE	243	207	210	208	179	169	168	170
Neural FSDE − rough Heston	−144	−136	−112	−104	−52	−44	−44	−38
Neural FSDE − Neural SDE	−150	−133	−121	−114	−76	−74	−74	−76

On SPY, the medians of the anchored Neural SDE and Neural FSDE and of the anchor-free Neural FSDE stay within 4 bps of their 2^{17}-path values from 2^{15} paths on, and those of rough Heston and PDV within 6 bps. Between 2^{16} and 2^{17} paths, the median of the anchor-free Neural SDE rises by 9 bps and that of rough Bergomi still falls by 20 bps. On SPXW the baselines converge more slowly, since the European chain contains many far-from-the-money quotes, while the Neural FSDE's median at 2^{15} paths is within 1 bp of its 2^{17}-path value.

All results in the paper use $B = 2^{15} = 32{,}768$ evaluation paths on both instruments. The Neural FSDE has the lowest median at every B on both instruments: at 2^{17} paths it is below rough Heston, rough Bergomi, PDV and both Neural SDEs on all 61 SPXW days, and below rough Heston on 56 of the 60 SPY days and below rough Bergomi, PDV and both Neural SDEs on all 60.

D.3 Optimisation Budget

All calibrations in Table 2 use 60 Adam steps. To check that the comparison is not an artefact of this budget, in particular that the classical models are not merely under-optimised, we repeated the full sweep with 200 steps for rough Bergomi, rough Heston and the Neural FSDE on all eight settings and every day, all else unchanged (Table 17). The extra budget lowers rough Bergomi's median by 100–257 bps, changes rough Heston's by at most 3 bps, and lowers the Neural FSDE's by 30–58 bps on every setting except SPXW (b), where it rises by 17. The Neural FSDE remains below both classical models on all eight settings, its margin over the better of them widens on seven of eight (from 19–128 to 49–155 bps), and all 16 paired comparisons at 200 steps have bootstrap intervals below zero. The Neural SDE was also rerun at 200 steps, but at the bound $c = 3$ rather than at its best bound, so we omit it here (Appendix D.4).

D.4 Diffusion Bound

Both neural models bound the diffusion of the log-variance $L = \log V$ as $\sigma_\phi = c \tanh(\cdot)$. We compare them first at the common bound $c = 3$, the Neural FSDE's best on the SPXW (a) pilot (Table 18), and then with each model at its own best bound. The Caputo scheme multiplies the diffusion by $w_0^\sigma = \Delta t^{\alpha - 1/2}/\Gamma(\alpha)$ in the one-step noise (Eq. 9), which at the learned $\hat{\alpha} \approx 0.64$ and $N = 128$ is about four times its value at $\alpha = 1$. A lower α therefore also raises the effective volatility of volatility, and if the Neural SDE's diffusion is held back by the bound, part of the memory effect, the gap between the Neural FSDE and the Neural SDE measured in Appendix D.5, is a scale effect instead. A diagnostic on all 483 SPXW (a) days confirms that the bound $c = 3$ holds the Neural

Table 17: Optimisation budget: median IV-RMSE (bps) after 60 and 200 Adam steps, each budget over the days common to its three models, and the Neural FSDE's margin over the better of rough Bergomi and rough Heston. All other settings as in Table 2.

		SPXW				SPY			
Model	Steps	(a)	(b)	(c)	(d)	(a)	(b)	(c)	(d)
rough Bergomi	60	585	727	407	537	443	448	384	508
	200	328	597	283	333	273	282	284	314
rough Heston	60	344	347	232	296	210	223	207	206
	200	345	347	232	296	208	220	205	205
Neural FSDE	60	216	269	212	212	165	182	175	187
	200	174	286	154	167	135	149	136	155
margin	60	−128	−78	−20	−85	−45	−41	−32	−19
	200	−155	−61	−78	−128	−73	−71	−70	−49

SDE's diffusion back. When we rebuild each day's fitted models and simulate $1{,}024$ paths, the Neural SDE's diffusion sits at the bound ($|\tanh| > 0.99$) on 99.2% of the states it visits (median $|\sigma_\phi| = 3.00$), against 59% for the Neural FSDE (median 2.98). On a third of the days (158 of 483), the Neural FSDE's diffusion is below the bound on almost all visited states. The one-step noise of the Neural FSDE is 4.0 times the Neural SDE's (10th–90th percentile over days 3.4–4.4), and matching it would require $|\sigma_\phi|$ of about 12 in the Neural SDE (median over days), above the bound on every day. The mechanism needs a fine time grid: the ratio of the fractional to the Markovian one-step noise weight is $\Delta t^{\alpha-1}/\Gamma(\alpha)$, which exceeds one only for $\Delta t < \Gamma(\alpha)^{-1/(1-\alpha)}$, a threshold of about 0.4 at $\alpha = 0.68$, with Δt in the model's time unit. On the COVID grid ($\Delta t = 1$ day) the ratio is $1/\Gamma(\hat{\chi}) \approx 0.75$ and the later weights decay, so a shared bound there limits the fractional model first and cannot inflate its advantage.

Selecting the bound per model. We recalibrated each neural model, in its anchored and anchor-free variants (Appendix D.5), with $c \in \{6, 12, 24\}$ on the 61-day subset (every eighth day) of SPXW (a), and the two Neural SDE variants also on that of SPY (a), with the same seeds and budget as the $c = 3$ runs, and give each model the one bound with the lowest median IV-RMSE over its pilot days. Table 18 shows SPXW (a). Both Neural SDE variants are best at $c = 12$, where their diffusion no longer saturates: the anchored one improves from 630 to 341 bps and the anchor-free one from 629 to 363. Both Neural FSDE variants behave the other way: they are best at $c = 3$ and degrade as the bound grows, while $\hat{\alpha}$ rises from 0.643 to about 0.68, so a common large bound would handicap them in turn. At every common bound the Neural FSDE remains ahead, by 89–422 bps anchored and 90–394 bps anchor-free. With each model at its best bound the gap is -133 bps anchored (95% interval $[-158, -119]$; the Neural FSDE is lower on 60 of 61 days) and -129 bps anchor-free ($[-155, -107]$; 61 of 61). Most of the gap under the common bound $c = 3$ is therefore a diffusion-scale effect, and the remainder, which persists with each model at its best bound, is the contribution of the fractional order. The Neural FSDE keeps $c = 3$ on SPY without a pilot there, which can only understate its accuracy. On the 61 SPY (a) pilot days the anchored Neural SDE likewise improves from 411 bps at $c = 3$ to 266 at $c = 6$ and 244 at $c = 12$, and the anchor-free one from 408 to 292 and 259 bps. The SPY $c = 24$ arms were stopped early, because $c = 24$ was already worse than $c = 12$ for both Neural SDE variants on SPXW (a).

Full population. Over all days of the eight settings, the anchored Neural SDE's median at its best bound $c = 12$ is 105–323 bps below its median at the common bound $c = 3$, and the Neural FSDE's lead shrinks from 174–436 to 69–132 bps (Table 15), with every 95% interval at least 60 bps below zero and the Neural FSDE lower on 435–480 days per setting. The bound therefore accounts for 60–74% of the gap at $c = 3$, and the fractional order for the rest. The anchor-free pair behaves alike: at $c = 12$ the anchor-free Neural SDE improves by 86–316 bps on its $c = 3$ calibration, and the anchor-free Neural FSDE's lead over it shrinks from 142–415 to 44–124 bps, so the bound accounts for 61–84% of that gap (Appendix D.5). As a stress test we also recalibrated the Neural FSDE at the Neural SDE's bound $c = 12$, which costs it 32–50 bps against its own bound. It still beats the Neural SDE on all eight settings, by 23–88 bps, with every interval at least 10 bps below zero and the Neural FSDE lower on 343–416 days per setting. At that bound it would no longer lead rough Heston everywhere: its median IV-RMSE is 13–30 bps above rough Heston's on SPXW (c) and SPY (d), within 8 bps of it on SPY (a)–(c), where the 95% intervals contain zero, and 46–84 bps below it

Table 18: Diffusion-bound pilot on the 61-day subset of SPXW (a): median IV-RMSE (bps) of the four neural models recalibrated with bound c (same seeds and budget; the $c = 3$ row uses the full-population calibrations), the Neural FSDE's learned $\hat{\alpha}$, the median over days of the share of visited states at which the anchored Neural SDE's diffusion sits at its bound ($|\tanh| > 0.99$), and the gap Neural FSDE minus Neural SDE at a common bound with its paired-bootstrap interval. Bold: each model's best bound. Last row: each model at its best bound. NSDE: Neural SDE.

	anchored			anchor-free			gap Neural FSDE − Neural SDE	
c	NSDE	Neural FSDE	$\hat{\alpha}$	NSDE	Neural FSDE	at bound	anchored	anchor-free
3	630	**207**	0.643	629	**234**	99.5%	-422 $[-451, -389]$	-394 $[-448, -359]$
6	389	229	0.671	395	248	94.5%	-161 $[-179, -132]$	-147 $[-197, -115]$
12	**341**	252	0.683	**363**	270	0.0%	-89 $[-118, -65]$	-93 $[-121, -63]$
24	388	262	0.681	391	301	0.0%	-125 $[-147, -74]$	-90 $[-119, -41]$
best							-133 $[-158, -119]$	-129 $[-155, -107]$

on SPXW (a), (b) and (d). Table 2 gives each model its own best bound, selected by the same rule from the same grid.

D.5 Memory × Anchor Factorial

The Neural FSDE's drift $\mu_\theta(L_t) + \lambda(\log \xi_0(t) - L_t)$ (Section 4.3) has two parts: a neural network μ_θ of the current log-variance, and a mean reversion, at a learnable rate λ, towards a learnable piecewise-constant log forward variance curve $\log \xi_0(t)$ with one value per maturity bin (under 30, 30 to 90, 90 to 180, and 180 days and over), initialised each day from the at-the-money implied variance in that bin. We call the second part the term-structure anchor. The anchor-free variant removes it, so its drift is $\mu_\theta(L_t)$ alone, with neither λ nor ξ_0. Everything else is unchanged: the initial log-variance (initialised each day from the at-the-money implied variance of the shortest maturities), the diffusion network and its bound, the correlation, the fractional order, and the calibration and evaluation protocol. Because μ_θ does not depend on time, the anchor-free model has no direct way to give different maturities different variance levels; the anchor provides one, as the forward variance curve does in rough Bergomi.

To separate the contribution of the fractional order from that of the anchor, we calibrate the anchor-free variants of both neural models under the identical cold-start protocol, on every setting and day, each at its best diffusion bound (Appendix D.4). With the anchored models of Table 2 this gives a 2×2 factorial over $\{\alpha < 1 \text{ learnable},\ \alpha = 1\} \times \{\text{anchored},\ \text{anchor-free}\}$ at matched architecture. Table 19 lists the four models' medians and Table 20 the four paired effects in the statistic of Appendix D.1. The fractional order remains the larger factor with or without the anchor: -44 to -124 bps without it and -69 to -132 bps with it, every interval below zero and every sign test at $p < 10^{-48}$. The anchor is a refinement that helps both models modestly: -4 to -18 bps on the memoryless model, with an interval below zero on six settings, and -22 to -32 bps on the fractional one, below zero on all eight. Its effect comes closest to that of memory on SPXW (c), where memory without the anchor gains 44 bps and the anchor given memory 32. At the common bound $c = 3$ the memory effects are larger, -142 to -415 and -174 to -436 bps, because that bound holds the memoryless model back.

Table 19: Median IV-RMSE (bps) of the four neural models of the factorial across all trading days, "all" quadrant, each at its best diffusion bound ($c = 3$ for the Neural FSDE, $c = 12$ for the Neural SDE). The anchored rows repeat Table 2.

	SPXW (European)				SPY (American)			
Model	(a)	(b)	(c)	(d)	(a)	(b)	(c)	(d)
Neural SDE, anchor-free	362	390	289	339	258	274	260	296
Neural FSDE, anchor-free	238	292	245	238	193	208	204	214
Neural SDE, anchored	348	382	285	328	244	260	244	278
Neural FSDE, anchored	216	269	212	212	165	182	175	187

Table 20: Memory × anchor factorial at matched architecture: paired effects on the median IV-RMSE (bps) in the statistic of Appendix D.1, $\mathrm{med}(A) - \mathrm{med}(B)$ over the days common to rough Bergomi, rough Heston and the four neural models of Table 19, each neural model at its best diffusion bound ($c = 3$ for the Neural FSDE, $c = 12$ for the Neural SDE), with paired-bootstrap 95% intervals (10,000 resamples). *Memory*: Neural FSDE (A) minus Neural SDE; *anchor*: anchored (A) minus anchor-free model. Negative values favour A. Stars: exact two-sided sign test on the per-day differences, $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

Setting	memory, anchor-free	memory, anchored	anchor, $\alpha = 1$	anchor, $\alpha < 1$
SPXW (a)	-124^{***} $[-134, -116]$	-132^{***} $[-140, -125]$	-14^{***} $[-22, -6]$	-22^{***} $[-27, -17]$
SPXW (b)	-98^{***} $[-112, -85]$	-113^{***} $[-126, -100]$	-8^{***} $[-20, +2]$	-23^{***} $[-33, -14]$
SPXW (c)	-44^{***} $[-51, -37]$	-73^{***} $[-80, -61]$	-4 $[-15, +3]$	-32^{***} $[-39, -26]$
SPXW (d)	-101^{***} $[-113, -93]$	-116^{***} $[-123, -107]$	-11^{***} $[-24, -6]$	-27^{***} $[-34, -21]$
SPY (a)	-64^{***} $[-69, -58]$	-79^{***} $[-84, -74]$	-13^{***} $[-18, -8]$	-28^{***} $[-32, -24]$
SPY (b)	-66^{***} $[-72, -59]$	-78^{***} $[-83, -72]$	-14^{***} $[-20, -8]$	-26^{***} $[-32, -21]$
SPY (c)	-55^{***} $[-66, -49]$	-69^{***} $[-78, -64]$	-16^{***} $[-25, -8]$	-29^{***} $[-36, -23]$
SPY (d)	-82^{***} $[-90, -75]$	-91^{***} $[-100, -83]$	-18^{***} $[-26, -9]$	-27^{***} $[-33, -21]$

D.6 The Learned Fractional Order

Figure 14 shows the Neural FSDE's per-day $\hat{\alpha}$ on the six settings of schedules (a), (c) and (d), and Table 21 summarises it together with the orders implied by the rough-Bergomi and rough-Heston fits. Across the 2,897 calibrations the learned order never approaches the memoryless end: no day exceeds 0.95, the per-setting medians lie in 0.638–0.675 with interquartile ranges near 0.03, and every daily estimate lies in $[0.586, 0.752]$ over the two-year window, including its high-volatility episodes. Three seeds on a 30-day replicate of SPXW (c) give per-seed medians of 0.6491, 0.6396 and 0.6394 (per-day range across seeds: median 0.022, maximum 0.074). The Neural FSDE's orders lie below the classical models' (0.70–0.71), so its fitted volatility is rougher.

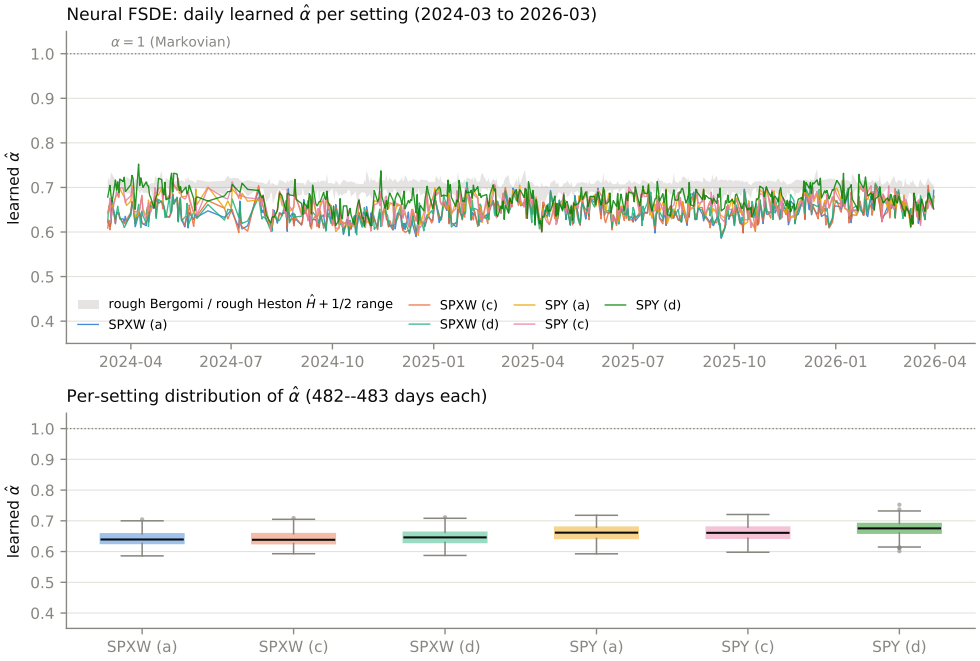


Figure 14: Per-day learned fractional order $\hat{\alpha}$ of the Neural FSDE on the six settings of schedules (a), (c) and (d) (482–483 trading days each; schedule (b) re-prices day t's calibration). Top: daily series, with the band spanned by the rough-Bergomi ($\hat{H} + \frac{1}{2}$) and rough-Heston fits on the same days. Bottom: per-setting distributions. The dotted line marks the memoryless $\alpha = 1$.

To test whether the fractional order α in the Neural FSDE is a useful learnable degree of freedom on the option data, or whether a fixed value would do as well, Table 22 compares the learnable-α

Table 21: Per-day learned fractional order on the six settings of schedules (a), (c) and (d): the Neural FSDE's $\hat{\alpha}$ (median, interquartile range, range, and days above 0.95) and the medians of the orders implied by the rough-Bergomi ($\hat{H} + \frac{1}{2}$) and rough-Heston fits on the same days.

	Neural FSDE $\hat{\alpha}$				rough Bergomi	rough Heston
Setting	days	median [IQR]	range	> 0.95	median	median
SPXW (a)	483	0.639 [0.627, 0.657]	[0.586, 0.705]	0	0.703	0.699
SPXW (c)	483	0.638 [0.626, 0.658]	[0.593, 0.709]	0	0.704	0.699
SPXW (d)	483	0.646 [0.630, 0.662]	[0.587, 0.712]	0	0.706	0.699
SPY (a)	482	0.661 [0.643, 0.679]	[0.593, 0.718]	0	0.709	0.700
SPY (c)	483	0.661 [0.643, 0.679]	[0.598, 0.721]	0	0.709	0.700
SPY (d)	483	0.675 [0.660, 0.691]	[0.601, 0.752]	0	0.710	0.700

Neural FSDE with calibrations in which α is frozen at one of five values across the admissible range, on three settings, SPXW (a), SPXW (c) and SPY (a), each on an evenly spaced subset of about 60 days. Architecture, hyperparameters and calibration loop are those of Section 4.3 (training details in Appendix D.2); only u_α is removed from the learnable parameters and clamped.

Table 22: Fixed-α ablation on three settings: median IV-RMSE (bps; interquartile range in brackets) with the Neural FSDE's fractional order frozen at five values, against the learnable-α Neural FSDE on the same days (evenly spaced subsets; the SPY subset keeps the 60 of its 61 days on which the learnable run exists). Same architecture and calibration protocol; only α is held fixed. Bold: lowest per column.

Configuration	SPXW (a), n=61	SPXW (c), n=61	SPY (a), n=60
fixed $\alpha = 0.55$	**198.5** [184.8, 223.8]	210.4 [162.6, 239.2]	**160.8** [137.7, 179.3]
fixed $\alpha = 0.65$	214.6 [188.3, 240.9]	**202.6** [168.2, 277.9]	165.6 [138.2, 195.3]
fixed $\alpha = 0.75$	305.8 [264.2, 383.1]	235.9 [191.5, 357.2]	181.3 [154.2, 236.2]
fixed $\alpha = 0.85$	443.5 [400.6, 525.0]	321.4 [257.5, 479.0]	246.8 [205.1, 295.2]
fixed $\alpha = 0.95$	572.1 [521.5, 662.8]	491.0 [360.4, 679.8]	351.7 [284.4, 437.3]
learnable α (Neural FSDE)	207.4 [193.6, 244.4]	208.9 [168.4, 268.7]	169.1 [143.0, 197.4]

The error rises steeply as α is frozen towards the memoryless end on all three settings: fixing $\alpha = 0.95$ costs 365, 282 and 183 bps relative to the learnable model, and the interquartile ranges at $\alpha = 0.95$ and under the learnable model do not overlap on any setting. The low end is a shallow basin: adjacent grid points on $[0.55, 0.65]$ differ by 5–16 bps, and the best fixed value is 9 bps below the learnable model on SPXW (a) ($\alpha = 0.55$), 6 bps on SPXW (c) ($\alpha = 0.65$) and 8 bps on SPY (a) ($\alpha = 0.55$). Learning α therefore finds this basin without per-setting tuning, at a cost of at most 9 bps against the best grid value. The grid stops at 0.55 because the model class requires $\alpha > \frac{1}{2}$ and the scheme's variance constant grows as $\mathcal{O}(1/(2\alpha - 1))$ towards that boundary (Appendix B.2).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract states three contributions, namely the Neural FSDE framework with a discrete adjoint, the first end-to-end real-data calibration of Bonyah's stochastic-fractional COVID-19 model, and an option-pricing benchmark on which the Neural FSDE attains the lowest median IV-RMSE on all eight (instrument, schedule) cells, with the gain from fractional memory measured against an architecture-matched Neural SDE given its best diffusion bound. Each is demonstrated in §3.3 (method) and §4.1–§4.3 (experiments).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: §3.2 notes that the discrete adjoint is specific to the Euler-Maruyama scheme, and §4.1 that it helps only the backward pass, with the forward still $\mathcal{O}(N^2)$ (costs in Appendix B); §4.2 shows that the fractional order is weakly identified when a neural drift acts on the five-dimensional COVID-19 state; §4.3 reports the shallow α basin, the per-model diffusion bound and ("Residual structure") where rough Heston still outperforms the Neural FSDE; Appendix D.4 shows that 60–74% of the memory gap at the common bound $c = 3$ is a diffusion-scale effect; and §5 lists the extensions left for future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The discrete adjoint identities are derived in §3.3 under the well-posedness assumption $\alpha \in (\frac{1}{2}, 1]$ from §3.1. The identities are verified symbolically by a sympy script in the released code (21/21 chain-rule identities pass) and numerically against PyTorch autograd to within $\sim 10^{-6}$ at five values of α in $[0.63, 0.98]$ (§4.1) and down to $\alpha = 0.55$ (Appendix B.5); the strong convergence of the scheme is checked in Appendix B.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: §3.1–§3.3 specify the Caputo FSDE forward and discrete adjoint completely; §4.1–§4.3 define datasets, splits, loss functions, and evaluation protocols; Appendices C.4 and D.2 list every training hyperparameter (architecture, optimiser, learning rates, batch sizes, time grid), and the further experiments of Appendices B, C.1 and D state their deviations from these settings. The released code provides the full PyTorch implementation and covers every experiment in the paper, including those in the appendices, with the commands that reproduce each result.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is available at `https://github.com/qmfin/NeuralFSDE`. OWID COVID-19 data is publicly available (URL given in §4.2). The S&P 500 option chains are sourced from WRDS, accessible to most university subscribers but not redistributable under the WRDS licence; the loader and filter code are released so that any WRDS subscriber can replicate the option-pricing experiments end-to-end.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All training and test details, including datasets, hold-out splits, optimiser, learning rates, batch sizes, time grid, loss functions, and evaluation metrics, are documented in §4.1–§4.3 and Appendices C.4 and D.2 (full hyperparameter listings for COVID-19 and option pricing respectively); the further analyses (Appendices A.1, B, C.1, C.5 and D) inherit the same settings unless they state otherwise.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Body Table 2 reports the median IV-RMSE across the 483 trading days per cell; per-day distributions appear as boxplots in Appendix D.1; Appendix D.1 reports paired-bootstrap 95% confidence intervals ($B = 10\,000$ replicates) and exact-binomial sign tests for the 32 (cell $\times$ comparator) differences, and Appendices D.5, D.3 and D.4 report the same statistics for the factorial, budget and diffusion-bound controls. For COVID-19 forecasting, body Table 1 reports the median over 48 squared-error observations (16 weekly origins $\times$ 3 random seeds) per (slice, channel) cell; Appendix C.1 reports small-sample-corrected Diebold-Mariano tests, origin-clustered bootstrap intervals and sign tests at the level of the 16 origins, with a Holm correction across cells. The toy α-recovery reports recovered $\hat{\alpha}$ across 15 runs (5 initialisations $\times$ 3 seeds) in Appendix A.1. Adjoint-autograd benchmark timings are the median over 3 (Figures 2–4) or 5 (Table 3) timed iterations after warm-up.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments ran on a single NVIDIA H100 GPU, about 200 GPU hours in total, including analyses the paper does not report; Appendices A, C.4 and D.2 give the size of each experiment (time grids, numbers of Monte Carlo paths, calibrations and optimisation steps).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses publicly available aggregated COVID-19 statistics (OWID) and end-of-day options data with no personally-identifiable information; baseline models and pretrained forecasters are properly cited and used under their licences.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The work is foundational methodology, a discrete adjoint method for Caputo fractional SDEs and a unified neural-fractional simulator. Potential positive impacts include more accurate volatility-surface calibration for option market-makers (§4.3) and a methodologically rigorous re-calibration of the Bonyah stochastic-fractional COVID-19 model (§4.2) that may inform future epidemiology work. The two real-world experiments compare against existing finance and epidemiology baselines without proposing any deployment, generative model, or system-level recommendation. No deployment is proposed and the framework is intended as a research tool rather than a production system; we note that misuse or overreliance on calibrated forecasts in epidemiology, or on calibrated pricers in finance, could nevertheless cause harm if they substitute for domain judgement.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases method code and per-day calibrators on aggregated public financial and epidemiological data. It does not release pre-trained generative models, scraped datasets, language models, or any high-misuse-risk asset.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All baseline models are cited (Bonyah et al. 2023 for the COVID-19 fSDE; Bayer et al. 2016 for rough Bergomi; El Euch & Rosenbaum 2019 for rough Heston; Guyon & Lekeufack 2023 for PDV; Kidger et al. 2020 for the Neural CDE; Bergerhausen et al. 2025 for the Neural SVE; Granger & Joyeux 1980 and Hosking 1981 for ARFIMA; Bjerksund & Stensland 2002 for the American IV inversion; Longstaff & Schwartz 2001 for LSM; PyTorch / Apache-2.0 for the framework). The pretrained Chronos and TimesFM forecasters are used under their Apache-2.0 licences. OWID COVID-19 data is publicly available under its terms (URL given in §4.2); S&P 500 option chains are obtained from WRDS under the standard subscriber licence and are not redistributed.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released code (`https://github.com/qmfin/NeuralFSDE`) ships with a `README.md` that describes the repository layout, environment (Python 3.10, PyTorch 2.5.1+cu121, CUDA 12.1; `uv sync` to install), and per-experiment reproduction commands.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper involves no crowdsourcing and no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper involves no research with human subjects; the COVID-19 data is aggregated population-level statistics from OWID with no personal identifiers.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology, a Caputo fractional SDE solver and its discrete adjoint, does not involve LLMs as any important, original, or non-standard component.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.