Score-Based Denoising for Semimartingales: Divergence-, Jump-, and Posterior/Oracle-Corrected Time Reversal

Louis Bouchard
(Dated: October 3, 2025)

We develop a score-based denoising theory for general Itô semimartingales that accommodates additive Gaussian noise, state-dependent multiplicative noise, and jump noise in one construction. The reverse-time dynamics follow the Föllmer–Haussmann–Pardoux representation and include two corrections often omitted in practice: a divergence term when the diffusion depends on state, and an exact reverse jump compensator written as a density ratio rather than a small-jump exponential approximation. We specialize to the geometric variance-preserving process and record the correct log-space drift with mean shift $-\Lambda_t$, which is required for speckle models. We also introduce posterior and semi-oracle score constructions for real data by adding likelihood gradients to the prior score, by using Tweedie and empirical-Bayes plug-ins in linear and log domains, by mapping denoisers to scores, and by employing Stein or score-matching estimators on unlabeled data. On the algorithmic side, we formulate probability-flow ODE sampling in log-SNR time with the proper coefficient $v = f - \nabla\cdot a - a\,s$ that matches forward SDE marginals only for continuous diffusions with $\nu \equiv 0$, we describe residualization around analytic scores, and we design hybrid Gaussian–jump integrators based on a score path-integral density-ratio estimator. We include practical guidance for stability, schedule calibration, and step efficiency on real instruments.

I. INTRODUCTION

Estimating latent clean signals from corrupted observations is a central task in imaging and spectroscopy. Classical approaches include linear minimum-variance estimators such as Wiener and Kalman filters, deterministic regularization such as total variation, wavelet thresholding, and sparsity, and Bayesian sampling of joint signal–noise models. Each family performs well under its own assumptions, but none is uniformly reliable when the noise is non-additive, heteroscedastic, or impulsive.

Score-based generative modeling learns the gradient of the log density, the score, across a continuum of noise scales and then integrates a reverse-time stochastic or deterministic dynamics to map a tractable reference distribution back to data [1]. In the additive, state-independent setting with

$$dX_t = f(X_t, t)dt + g(t)dW_t, \tag{1}$$

Anderson's time-reversal formula gives the backward drift $f - g^2\nabla_x \log p_t$ [4]. This model captures thermal readout and Johnson–Nyquist noise, but many instruments violate its assumptions.

Real measurements often exhibit state-dependent fluctuations such as speckle in coherent modalities, heavy-tailed non-Gaussian behavior, and discontinuous paths from impulsive events such as cosmic rays. A general treatment must therefore admit diffusion with x-dependent coefficients and jumps, and it must support sensor-aware sampling on real data when neither the ground truth nor the exact noise law is fully known.

We present a self-contained development of score-based denoising for Itô semimartingales with continuous and jump parts, and we align implementation with stochastic analysis. First, we state the Föllmer–Haussmann–Pardoux time-reversal for general Itô semimartingales and make explicit the divergence correction for state-dependent diffusion and the exact reverse jump compensator written as a density ratio $\nu^\star(x,t;dz) = \nu(x-z,t;dz)\,p_t(x-z)/p_t(x)$ [2, 10]. Second, we specialize to the geometric variance-preserving SDE $dX_t = -\frac{1}{2}\beta(t)X_t dt + \sqrt{\beta(t)}X_t dW_t$, record the correct log transform $d(\log X_t) = -\beta(t)dt + \sqrt{\beta(t)}dW_t$, and derive the associated reverse drift including the divergence term. Third, we introduce posterior and semi-oracle score constructions suited to real data, obtained by adding likelihood gradients to the prior score, by using Tweedie and empirical-Bayes plug-ins in linear and log domains, by mapping denoisers to scores, and by using Stein or score-matching estimators [5, 6]; we treat these as baselines and learn small residuals on top. Fourth, we give probability-flow ODE samplers in log-SNR time with embedded error control, we use residualization by analytic scores, and we design hybrid Gaussian–jump integrators based on a score path-integral approximation to the density ratio; we clarify that PF–ODE marginals match the forward SDE only for continuous diffusions with $\nu \equiv 0$ [1, 12]. Finally, we connect the construction to classical estimators by showing how Wiener-type estimators and homomorphic log-domain despeckling arise as special cases of the posterior and residualized formulations.

We work on $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ with usual conditions. Vectors are columns; x_i and $x^\top$ denote the ith entry and the transpose. The Euclidean norm is $\|x\|_2$. For a matrix A, the Frobenius norm is

$\|A\|_{\mathrm{F}} = \sqrt{\mathrm{Tr}(A^\top A)}$. We write I_d for the $d \times d$ identity. Unless stated, equalities hold $\mathbb{P}$–almost surely. Gradients act row-wise so that $\nabla_x \log p(x) \in \mathbb{R}^d$ for scalar p. The Hadamard product uses the symbol $\odot$. We write $a(x,t) = \frac{1}{2}\Sigma\Sigma^\top(x,t)$, and all stochastic integrals are Itô.

II. FORWARD DIFFUSION MODEL

A. Canonical path space and semimartingale model

Let $d \in \mathbb{N}$ be the ambient dimension. We work on the canonical Skorokhod space $D([0,T],\mathbb{R}^d)$ with Borel σ-algebra $\mathcal{D}$ and right-continuous canonical filtration $\{\mathcal{D}_t\}_{t\in[0,T]}$ generated by $X_t(\omega) = \omega(t)$. A probability measure $\mathbb{P}_X$ on $(D,\mathcal{D})$ solves the martingale problem for the Itô semimartingale

$$\begin{aligned} dX_t = f(X_{t-},t)\,dt + \Sigma(X_{t-},t)\,dW_t \\ + \int_{\|z\|>0} z\,\widetilde{N}(dt,dz), \end{aligned} \tag{2}$$

if the following hold: W is a standard $\mathbb{R}^d$-valued Brownian motion; $N(dt,dz)$ is a Poisson random measure on $(0,T]\times(\mathbb{R}^d\setminus\{0\})$ with compensator $\nu(X_{t-},t;dz)\,dt$, independent of W; $\widetilde{N}(dt,dz) = N(dt,dz) - \nu(X_{t-},t;dz)\,dt$; and the usual integrability conditions are satisfied. Equation (2) collects the continuous Brownian part and the jump part; setting individual terms to zero recovers the diffusive or pure-jump limits. Superscripts $i,j \in \{1,\dots,d\}$ refer to Cartesian components. Einstein summation is not used.

It is convenient to record the predictable characteristics of X for the truncation $h(z) = z\,\mathbb{1}_{\{\|z\|<1\}}$:

$$\begin{aligned} B_t &= \int_0^t f(X_{s-},s)\,ds, \\ C_t &= \int_0^t \Sigma\Sigma^\top(X_{s-},s)\,ds, \\ \nu^X(\omega;ds,dz) &= \nu(X_{s-}(\omega),s;dz)\,ds. \end{aligned} \tag{3}$$

These quantities will be used in the reverse-time representation.

B. Coefficient hypotheses

Drift f. Assume $f:\mathbb{R}^d\times[0,T]\to\mathbb{R}^d$ is Borel, locally Lipschitz in x, piecewise $\mathcal{C}^1$ in t, and of at most linear growth:

$$\begin{aligned} \|f(x,t)-f(y,t)\|_2 &\le L_f\,\|x-y\|_2, \\ \|f(x,t)\|_2 &\le K_f\,(1+\|x\|_2). \end{aligned} \tag{4}$$

Diffusion Σ. Let $\Sigma:\mathbb{R}^d\times[0,T]\to\mathbb{R}^{d\times d}$ be jointly Borel, locally Lipschitz in x, and of at most linear growth:

$$\begin{aligned} \|\Sigma(x,t)-\Sigma(y,t)\|_{\mathrm{F}} &\le L_\Sigma\,\|x-y\|_2, \\ \|\Sigma(x,t)\|_{\mathrm{F}} &\le K_\Sigma\,(1+\|x\|_2). \end{aligned} \tag{5}$$

When needed for absolute continuity of time marginals, impose uniform ellipticity on an open state domain $\mathsf{D}\subseteq\mathbb{R}^d$:

$$\xi^\top\big(\Sigma\Sigma^\top\big)(x,t)\,\xi \ge \lambda_{\min}\,\|\xi\|_2^2, \tag{6}$$

with $\lambda_{\min} > 0$. For multiplicative models such as $\Sigma(x,t) = g(t)\,\mathrm{diag}(x)$ one may restrict to $\mathsf{D} = (0,\infty)^d$ or add a small stabilizer $x\mapsto x+\varepsilon$; see the multiplicative section.

Jump kernel ν. Let

$$\nu:\mathbb{R}^d\times[0,T]\to\big\{\text{Radon measures on } \mathbb{R}^d\setminus\{0\}\big\} \tag{7}$$

be predictable in (t,ω) through X_{t-} and Borel in x. Assume local Lipschitz continuity in x in the sense of boundedness on compacts and the Lévy-type growth bound

$$\int_{\|z\|>0}\big(\|z\|_2^2\wedge 1\big)\,\nu(x,t;dz) \le K_\nu\big(1+\|x\|_2^2\big), \tag{8}$$

which ensures that $\int z\,\widetilde{N}(dt,dz)$ is a square-integrable martingale. A constant kernel recovers compound Poisson jumps with rate λ and mark law $\mu(dz) = \nu(dz)/\lambda$. Both finite- and infinite-activity regimes are included by (8).

Under (4)–(8), with W independent of N and $\mathbb{E}\|X_0\|_2^2 < \infty$, the SDE (2) admits a unique strong, nonexplosive solution; see Theorem IX.2.31 of [7]. When (6) holds on D, the time marginals admit densities p_t that are strictly positive and smooth on D. For the jump case, this holds under standard nondegeneracy of the compound kernel. This absolute continuity will be assumed whenever scores $\nabla_x \log p_t$ are used later.

C. Infinitesimal generator and Kolmogorov equation

Set $a(x,t) = \frac{1}{2}\,\Sigma\Sigma^\top(x,t)$. For $\varphi\in\mathcal{C}_c^2(\mathbb{R}^d)$, the time-inhomogeneous generator acting on the core test space is

$$(\mathcal{L}_t\varphi)(x) = f^i(x,t)\,\partial_{x_i}\varphi(x) + a^{ij}(x,t)\,\partial^2_{x_i x_j}\varphi(x) + \int_{\|z\|>0}\Big[\varphi(x+z) - \varphi(x) - \mathbb{1}_{\{\|z\|<1\}}\, z^\top \nabla\varphi(x)\Big]\,\nu(x,t;dz), \tag{9}$$

with the convention that repeated indices are not summed unless an explicit summation sign is present.

Kolmogorov forward equation (weak form). Let $\{p_t\}_{t\in[0,T]}$ be a family of probability densities on $\mathbb{R}^d$ with p_0 equal to the law of X_0. The Fokker–Planck equation is understood in the weak sense:

$$\frac{d}{dt}\int_{\mathbb{R}^d}\varphi(x)\,p_t(x)\,dx = \int_{\mathbb{R}^d}(\mathcal{L}_t\varphi)(x)\,p_t(x)\,dx, \quad \text{for all } \varphi\in\mathcal{C}^2_c(\mathbb{R}^d). \tag{10}$$

Formally,

$$\partial_t p_t = \mathcal{L}_t^* p_t, \quad p_{t=0} = p_0. \tag{11}$$

When ν is x-independent, the jump term admits a strong form and (11) becomes the integro-differential PDE

$$\partial_t p_t(x) = -\nabla\cdot\big(f(x,t)\,p_t(x)\big) + \partial_{x_i}\partial_{x_j}\big(a^{ij}(x,t)\,p_t(x)\big) + \int_{\|z\|>0}\Big[p_t(x-z) - p_t(x) + \mathbb{1}_{\{\|z\|<1\}}\, z^\top\nabla p_t(x)\Big]\,\nu(t;dz). \tag{12}$$

In the general x-dependent case the forward operator involves a density-ratio weighting inside the jump integral, and we use only the weak form (10) to avoid unnecessary regularity assumptions. Under the hypotheses in Section II, and in particular under uniform ellipticity (6) on an open state domain, the time marginals of X_t are absolutely continuous and p_t is smooth on that domain.

D. Score and Fisher information

Score. Assume p_t is strictly positive and smooth enough that $\nabla_x p_t \in L^2(\mathbb{R}^d)$. The time-$t$ score is

$$\mathbf{s}_t(x) := \nabla_x \log p_t(x) = \frac{\nabla_x p_t(x)}{p_t(x)} \in \mathbb{R}^d. \tag{13}$$

It satisfies $\mathbb{E}_{p_t}[\mathbf{s}_t(X_t)] = 0$ and the Stein identity $\mathbb{E}_{p_t}[\nabla\cdot\phi(X_t) + \phi(X_t)^\top \mathbf{s}_t(X_t)] = 0$ for suitable test fields ϕ.

Fisher information. The scalar Fisher information at time t is

$$\mathcal{I}(p_t) := \mathbb{E}\big[\|\mathbf{s}_t(X_t)\|_2^2\big] = \int_{\mathbb{R}^d}\|\nabla_x \log p_t(x)\|_2^2\,p_t(x)\,dx. \tag{14}$$

For a Gaussian $p_t = \mathcal{N}(m_t, \sigma_t^2 I_d)$ one has $\mathcal{I}(p_t) = d/\sigma_t^2$. Under variance-preserving dynamics with $\sigma_t^2 \equiv 1$ this gives $\mathcal{I}(p_t) = d$.

Tweedie and empirical-Bayes identities. These identities connect the score to posterior means and provide plug-in oracle targets on real data. In the additive Gaussian case with $X_t = \bar\alpha_t X_0 + \sigma_t\epsilon$ and $\epsilon\sim\mathcal{N}(0,I)$,

$$\mathbb{E}[X_0 \mid X_t = x] = \frac{1}{\bar\alpha_t}\big(x + \sigma_t^2\,\mathbf{s}_t(x)\big), \qquad \mathbf{s}_t(x) = \frac{\bar\alpha_t\,\mathbb{E}[X_0\mid X_t = x] - x}{\sigma_t^2}. \tag{15}$$

Setting $\bar\alpha_t = 1$ recovers $\mathbb{E}[X_0 \mid X_t = x] = x + \sigma_t^2\,\mathbf{s}_t(x)$. In the multiplicative geometric VP case, let $Y_t = \log X_t$ and let $\mathrm{Var}(Y_t \mid Y_0) = \Lambda_t$. Then

$$\mathbb{E}[Y_0 \mid Y_t = y] = y + \Lambda_t\,\partial_y \log p_t^Y(y), \qquad \mathbf{s}_t(x) = \frac{1}{x}\,\partial_y \log p_t^Y(\log x) - \frac{1}{x}. \tag{16}$$

Formulas (15)–(16) motivate semi-oracle constructions used later, including posterior scores, denoiser-to-score mappings, and log-domain plug-ins for multiplicative noise.

E. Catalogue of physical noise models

Table I lists, for each physical mechanism, a specific choice of (f, Σ, ν) that matches its first-order

TABLE I. Representative noise sources and one set of coefficients that reproduces their salient statistics within the semimartingale model (2). "VP" denotes variance-preserving.

Origin	$f(x,t)$	$\Sigma(x,t)$	ν
Thermal / readout (VP)	$-\frac{1}{2}\beta(t)\,x$	$\sqrt{\beta(t)}\,I_d$	0
Speckle / SAR (geometric VP)	$-\frac{1}{2}\beta(t)\,x$	$\sqrt{\beta(t)}\,\mathrm{diag}(x)$	0
$1/f$ (colored)	$-\frac{1}{2}\beta(t)\,x$	augment with OU cascade; colored noise via auxiliary state	
Cosmic rays / impulsive	0	$g(t)\,I_d$	$\lambda\,\mu(dz)$
Shot (Poisson counts)	variance-stabilize (e.g., Anscombe) $\Rightarrow$ additive Gaussian in transformed domain		
Heavy-tailed outliers	$-\frac{1}{2}\beta(t)\,x$	$\sqrt{\beta(t)}\,I_d$	$\lambda\,\mu_\alpha(dz)$ (α-stable-type)

statistics. Speckle in coherent imaging is multiplicative, and $\Sigma(x,t)=\sqrt{\beta(t)}\,\mathrm{diag}(x)$ produces log-normal marginals on $x>0$. Long-memory $1/f$ spectra can be handled within the semimartingale framework by state augmentation using a cascade of Ornstein–Uhlenbeck filters that approximate the colored spectrum; fractional Brownian drivers are not Itô semimartingales and are therefore avoided here. Impulsive disturbances such as cosmic rays are captured by a compound Poisson kernel with rate λ and mark distribution μ. Shot noise can be mapped to an additive Gaussian regime by variance-stabilizing transforms. Heavy-tailed contamination may be modeled by Lévy measures with α-stable-type behavior, with infinite activity allowed under (8).

F. Worked examples

Example II.1 (Additive variance-preserving diffusion)**.** Let $f(x,t)=-\frac{1}{2}\beta(t)x$, $\Sigma(x,t)=g(t)I_d$ with $g(t)=\sqrt{\beta(t)}$, and $\nu\equiv 0$. Then

$$X_t = e^{-\frac{1}{2}\Lambda_t}X_0 + \int_0^t e^{-\frac{1}{2}(\Lambda_t-\Lambda_s)}\sqrt{\beta(s)}dW_s,$$
$$\Lambda_t = \int_0^t \beta(s)ds. \tag{17}$$

If $X_0\sim\mathcal{N}(0,I_d)$, then $X_t\sim\mathcal{N}(0,I_d)$ for all t. The score is $\mathbf{s}_t(x)=-x$. □

Example II.2 (Geometric variance-preserving diffusion)**.** Let $f(x,t)=-\frac{1}{2}\beta(t)x$, $\Sigma(x,t)=\sqrt{\beta(t)}\mathrm{diag}(x)$, and $\nu\equiv 0$. For $x>0$, set $Y_t=\log X_t$. Itô's lemma gives

$$dY_t = -\beta(t)dt + \sqrt{\beta(t)}dW_t,$$
$$Y_t\sim\mathcal{N}\big(Y_0-\Lambda_t,\Lambda_t I_d\big).$$

Hence X_t is log-normal with density

$$p_t(x)=\prod_{i=1}^d \frac{1}{x_i\sqrt{2\pi\Lambda_t}}\exp\Big(-\frac{(\log x_i-\mu_{t,i})^2}{2\Lambda_t}\Big), \tag{18}$$

$$\mu_{t,i}=(Y_0)_i-\Lambda_t.$$

Differentiation yields the analytic score

$$\mathbf{s}_t(x)=-\frac{\log x-\mu_t}{\Lambda_t}\odot\frac{1}{x}-\frac{1}{x}, \tag{19}$$

which we will use to stabilize training and to build log-space samplers. □

Example II.3 (Jump-diffusion with Poisson outliers)**.** Let $f\equiv 0$, $\Sigma(x,t)=g(t)I_d$, and $\nu(dz)=\lambda\mu(dz)$ with $\lambda>0$ and μ having a bounded density and finite second moment. Then

$$X_t = X_0+\int_0^t g(s)dW_s+\sum_{k=1}^{N_t}Z_k,$$
$$N_t\sim\mathrm{Poisson}(\lambda t),$$
$$Z_k\overset{\text{i.i.d.}}{\sim}\mu.$$

The jump component governs the tails of p_t. Under the stated conditions the law of X_t is absolutely continuous with a smooth density [9, Th. 27.7]. □

G. Why forward diffusion?

The representation of noisy dynamics by a triple (f,Σ,ν) is not unique, because different choices may yield identical marginals at the terminal time T. For score-based denoising, the specific pathwise decomposition is less important than two properties. First,

absolute continuity: the marginal laws p_t should admit smooth densities with respect to Lebesgue measure so that the score $\nabla_x \log p_t$ is well defined. This typically requires ellipticity in the diffusion part or nondegenerate jump kernels. Second, ergodicity to a tractable reference: the forward dynamics should converge to a distribution p_T that is analytically tractable and easy to sample. In practice one enforces variance-preserving schedules so that p_T is standard Gaussian for additive noise or log-normal for multiplicative noise, both of which can be sampled exactly and therefore anchor reverse-time integration.

These conditions explain why forward diffusion is used as a design element: it guarantees the existence of scores across noise scales and ensures that the terminal distribution is simple enough to start reverse-time integration from. Other triples (f, Σ, ν) with the same marginals are possible, but without these properties the reverse-time construction becomes ill posed or algorithmically impractical.

Having specified the forward dynamics and their statistical structure, we now turn to the reverse-time representation that underlies score-based denoising and where divergence and jump corrections play a decisive role.

III. REVERSE–TIME REPRESENTATION

The key observation behind score-based denoising is that one can run the dynamics backward in time if the drift is corrected by a term involving the score of the time-marginal density. We give a rigorous statement for Itô semimartingales that satisfy the hypotheses of Section II. Throughout, $T > 0$ is fixed, (f, Σ, ν) obey (4)–(8), and we set $a(x,t) := \frac{1}{2}\Sigma\Sigma^\top(x,t)$. For a matrix field a, write $(\nabla \cdot a)_j := \sum_{i=1}^d \partial_{x_i} a^{ij}$, meaning the divergence is taken column-wise.

A. Preliminaries on time reversal

For $\omega \in D([0,T], \mathbb{R}^d)$ define its time reversal by

$$\rho_T(\omega)(t) = \omega(T) - \omega(T-t-), \quad 0 \le t < T,$$
$$\rho_T(\omega)(T) = \omega(0).$$

Write $\bar{X}_t := X_{T-t-}$ for the reversed process on $[0,T]$ and let $\{\bar{\mathcal{F}}_t\}$ be its right-continuous natural filtration. A backward Brownian motion $\bar{W}$ is an $\mathbb{R}^d$-valued $\{\bar{\mathcal{F}}_t\}$-Brownian motion defined by $\bar{W}_t := W_T - W_{T-t}$. The time reversal of the compensated Poisson random measure is denoted by $\tilde{\bar{N}}$ [7, Ch. V.8].

B. Föllmer–Haussmann–Pardoux with jumps

Theorem III.1 (Reverse time for jump–diffusions)**.** *Assume $f, \Sigma \in \mathcal{C}^{1,2}$ with Σ full rank on the state domain and ν satisfies (8). Let p_t be a strictly positive classical solution of the Kolmogorov forward equation (11) with $p_t \in \mathcal{C}^{1,2}$ in (x,t). Define the score $\mathbf{s}_t(x) := \nabla_x \log p_t(x)$. Then the time-reversed process $\{\bar{X}_t\}_{t\in[0,T]}$ is a $\{\bar{\mathcal{F}}_t\}$-semimartingale with characteristics*

$$d\bar{X}_t = \big[f - \nabla\cdot a - 2a\mathbf{s}_t\big](\bar{X}_t, t)\,dt + \Sigma(\bar{X}_t, t)\,d\bar{W}_t + \int_{\|z\|>0} z\,\tilde{\bar{N}}(dt, dz), \tag{20}$$

where the compensated jump measure $\tilde{\bar{N}}$ is taken with respect to the reversed compensator

$$\nu^\star(x,t;dz)\,dt := \nu(x-z,t;dz)\,\frac{p_t(x-z)}{p_t(x)}\,dt. \tag{21}$$

The truncation function is the same as in the forward dynamics. In particular, for x-independent $\nu(dz)$ the reversal reduces to $\nu^\star(x,t;dz) = \nu(dz)\,p_t(x-z)/p_t(x)$.

Remark III.2 (Density ratio versus exponential tilt)**.** Equation (21) is exact. The exponential form $\exp\{-\mathbf{s}_t(x)^\top z\}$ sometimes used in the literature is a small-jump approximation obtained by a first-order Taylor expansion of $\log\big(p_t(x-z)/p_t(x)\big)$. It is accurate only when $\|z\|$ is small.

Sketch. We outline the main steps; see [2] for diffusions and [10] for jumps.

(i) Semimartingale characteristics. Write the forward X in its canonical form with truncation $h(z) = z\,\mathbb{1}_{\{\|z\|<1\}}$ and characteristics (B, C, ν^X), where $B_t = \int_0^t f(X_{s-}, s)\,ds$, $C_t = \int_0^t \Sigma\Sigma^\top(X_{s-}, s)\,ds$, and $\nu^X(\omega; ds, dz) = \nu(X_{s-}(\omega), s; dz)\,ds$.

(ii) Change of measure for the continuous martingale part. Introduce the density process

$$\mathcal{E}_t = \exp\Big(\int_0^t \mathbf{s}_s(X_s)^\top \Sigma(X_s,s)\, dW_s - \tfrac{1}{2}\int_0^t \|\Sigma^\top(X_s,s)\mathbf{s}_s(X_s)\|_2^2\, ds\Big),$$

and assume the usual integrability so that $\mathcal{E}_t$ is a martingale. Under the tilted measure, the continuous characteristics of the reversed process acquire the drift correction $-\nabla\cdot a - 2a\mathbf{s}_t$ [2].

(iii) Bayes formula for the jump compensator. For the jump part, identify the reversed predictable compensator by Bayes' rule on path space, which yields (21) [10, Th. 3.7]. Writing the jump integral with respect to the compensated measure $\tilde{\tilde{N}}$ associated with $\nu^\star$ gives (20) with the same truncation convention. □

C. Specializations and sanity checks

Additive diffusion (Anderson drift). If Σ is x-independent and $\nu \equiv 0$, then $a(t) = \frac{1}{2}\Sigma\Sigma^\top(t)$ and $\nabla\cdot a = 0$, so

$$d\bar{X}_t = \big[f - \Sigma\Sigma^\top \mathbf{s}_t\big](\bar{X}_t, t)\, dt + \Sigma(t)\, d\bar{W}_t, \tag{22}$$

in agreement with Anderson [4]. The divergence and jump corrections vanish in the state-independent Gaussian setting used by most diffusion models.

Geometric VP diffusion. With $\Sigma(x,t) = g(t)\,\mathrm{diag}(x)$, one has $a^{ij}(x,t) = \frac{1}{2}g^2(t)\,x_i^2\delta_{ij}$ and $\partial_{x_i}a^{ij} = g^2(t)\,x_j$. Substituting into (20) yields

$$\begin{aligned} d\bar{X}_t = \Big[-\tfrac{1}{2}\beta(t)\,\bar{X}_t - g^2(t)\,\bar{X}_t - g^2(t)\,\mathrm{diag}\big(\bar{X}_t^{\odot 2}\big)\,\mathbf{s}_t(\bar{X}_t,t)\Big]\, dt \\ + g(t)\,\mathrm{diag}(\bar{X}_t)\, d\bar{W}_t, \end{aligned} \tag{23}$$

which agrees with the specialization obtained by inserting the analytic log-normal score (19).

D. Probability–flow ODE

Define the probability–flow vector field

$$v_t(x) := f(x,t) - \nabla\cdot a(x,t) - a(x,t)\,\mathbf{s}_t(x). \tag{24}$$

When $\nu \equiv 0$ and Σ is nondegenerate, the forward SDE (2) and the reverse SDE (20) share the same time marginals as the deterministic ODE

$$\frac{dX_t^{\mathrm{PF}}}{dt} = v_t\big(X_t^{\mathrm{PF}}\big), \quad X_0^{\mathrm{PF}} = X_0, \tag{25}$$

as noted in [1]. Consequently, one may replace stochastic integration by higher-order ODE solvers (for example Heun or DPM–Solver) without altering the final distribution, although the path distribution does differ [12]. This PF–ODE equivalence does not extend to jump–diffusions with $\nu \neq 0$.

E. Existence and uniqueness of the reverse SDE

Proposition III.3. *Assume the forward coefficients satisfy (4)–(8) and that $\mathbf{s}_t$ is locally Lipschitz with at most linear growth on the state domain where $p_t > 0$. Then the reverse SDE (20) admits a unique strong solution on $[0,T]$.*

Proof. Local Lipschitz continuity of the reverse drift $v_t = f - \nabla\cdot a - 2a\,\mathbf{s}_t$ together with linear growth of $a\,\mathbf{s}_t$ ensures Yamada–Watanabe-type conditions for drift and diffusion. Nondegeneracy of Σ yields strictly positive quadratic variation for the continuous martingale part, and (8) gives finite quadratic variation for the jump martingale. A fixed-point argument gives strong existence, and pathwise uniqueness follows from Gronwall's inequality. □

F. Connection to Schrödinger bridges

The reverse-time construction admits an entropic interpolation viewpoint. Let $\mathbb{R}$ denote the reference

path law induced by the forward SDE

$$dX_t = f(X_t, t)dt + \Sigma(X_t, t)dW_t,$$

and let $\mathbb{Q}$ range over path measures on $D([0,T], \mathbb{R}^d)$ with prescribed end-point marginals (μ_0, μ_T). The Schrödinger bridge problem seeks

$$\min_{\mathbb{Q}\in\mathcal{P}} \mathrm{KL}(\mathbb{Q}\|\mathbb{R}) \quad \text{subject to} \quad \mathbb{Q}_{t=0} = \mu_0, \ \mathbb{Q}_{t=T} = \mu_T,$$

which in a dynamic Benamou–Brenier form is equivalent to minimizing a kinetic control cost under the Fokker–Planck constraint. Writing the controlled drift as $f + au_t$ with $a = \frac{1}{2}\Sigma\Sigma^\top$, the action reads

$$\int_0^T \frac{1}{2}\,\mathbb{E}_{\mathbb{Q}}\Big[\|u_t(X_t)\|^2_{a^{-1}(X_t,t)}\Big]\,dt, \tag{26}$$

subject to

$$\partial_t p_t = \mathcal{L}_t^* p_t - \nabla\cdot\big(au_t p_t\big), \tag{27}$$

where $\|v\|^2_{a^{-1}} = v^\top a^{-1} v$. The optimal control $u_t^\star$ is a gradient field given by Schrödinger bridge potentials, and the induced probability–flow drift

$$v_t(x) = f(x,t) - \nabla\cdot a(x,t) - a(x,t)\,\nabla\log p_t(x) \tag{28}$$

coincides with the score-corrected drift used in the probability–flow ODE. Thus the score term acts as a Lagrange multiplier enforcing the marginal constraints along the entropy-regularized interpolation between μ_0 and μ_T [13].

IV. SCORE ESTIMATION

Having identified the reverse-time drift (20) as a functional of the score $\mathbf{s}_t = \nabla_x \log p_t$, we now address how to learn $\mathbf{s}_t$ from data. In most applications the forward coefficients (f, Σ, ν) and the initial density p_0 are not known in closed form; we are given only an i.i.d. training set $\{x_0^{(n)}\}_{n=1}^{N_{\text{train}}} \sim p_0$. Denoising score matching (DSM) provides an unbiased and tractable surrogate for the intractable risk $\mathbb{E}_{p_t}\big[\|s_\theta - \mathbf{s}_t\|_2^2\big]$. We first state the continuous-time objective and then pass to a discrete time grid suitable for numerics. We conclude with choices of time weights and parameterizations that stabilize learning in high dimensions.

A. Continuous-time denoising score matching

Let $s_\theta : \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$ be a parameterized vector field, differentiable in x and measurable in t. The integrated DSM risk is

$$\mathcal{J}(\theta) = \int_0^T \alpha(t)\,\mathbb{E}_{x_0\sim p_0}\,\mathbb{E}_{X_t|x_0}\big[\|s_\theta(X_t,t) - \mathbf{s}_t(X_t)\|_2^2\big]\,dt, \tag{29}$$

where $\alpha : [0,T] \to (0,\infty)$ is a time-importance weight that compensates for the unequal difficulty of estimating $\mathbf{s}_t$ across noise scales. Typical choices are $\alpha(t) = \sigma_t^2$ for variance-preserving diffusions or the equal-time weight $\alpha(t) = 1$. The inner conditional expectation is with respect to the forward process (2) initialized at x_0, simulated offline by an SDE solver.

Remark IV.1 (Hyvärinen identity vs. DSM)**.** For sufficiently regular densities on $\mathbb{R}^d$, the Hyvärinen score $\int \|\nabla_x \log p_\theta - \nabla_x \log p_t\|_2^2\,dx$ is minimized by the true density p_t. DSM rewrites this risk in terms of noisy samples X_t and avoids evaluating p_θ explicitly, using the Stein identity $\mathbb{E}_{p_t}[\nabla_x \log p_t] = 0$ [5, 6]. Under mild conditions, the population minimizer satisfies $s_\theta^\star(\cdot,t) = \mathbf{s}_t(\cdot)$ almost everywhere.

Time parameterization and weights. For variance-preserving schedules it is advantageous to parameterize time by the log-SNR

$$\lambda(t) = \log\left(\frac{\bar\alpha_t}{\sigma_t}\right), \quad \bar\alpha_t = \exp\left(-\frac{1}{2}\Lambda_t\right), \quad \sigma_t = \sqrt{1-\bar\alpha_t^2}.$$

Sampling t uniformly in λ allocates training mass where the dynamics are stiff. A robust choice is

$$\alpha(t) = \big(\mathrm{SNR}(t)\big)^\gamma = \left(\frac{\bar\alpha_t}{\sigma_t}\right)^\gamma, \quad \gamma \in [1,2], \tag{30}$$

which equalizes gradient magnitudes across noise levels. For multiplicative models we use the log-domain analog with Λ_t in place of σ_t^2.

Semi-oracle residualization. On real data we often possess a semi-oracle score s_{semi} (posterior scores, Tweedie or empirical-Bayes plug-ins, or denoiser-to-score mappings; see Section III). We then learn only the residual

$$r_\theta(x,t) = s_\theta(x,t) - s_{\text{semi}}(x,t), \tag{31}$$

$$\min_\theta \ \mathbb{E}\big[\alpha(t)\,\|r_\theta(X_t,t) - (\mathbf{s}_t - s_{\text{semi}})(X_t,t)\|_2^2\big],$$

which reduces variance and improves stability.

B. Discretization in time

Numerical samplers operate on a finite grid $0 = t_0 < t_1 < \cdots < t_N = T$. We approximate (29) by

the Riemann sum

$$\mathcal{L}(\theta) = \frac{1}{N+1}\sum_{k=0}^{N}\mathbb{E}\Big[\lambda_k \big\| s_\theta(X_{t_k}, t_k) - \mathbf{s}_{t_k}(X_{t_k})\big\|_2^2\Big], \tag{32}$$

where $\lambda_k = \alpha(t_k)\,\Delta t_k/\bar{\alpha}$, $\Delta t_k = t_{k+1} - t_k$, and $\bar{\alpha} = \sum_{j=0}^{N}\alpha(t_j)\Delta t_j$. In practice we estimate (32) with a single Monte Carlo draw per optimizer step by sampling $x_0 \sim p_0$ and simulating the forward noise to obtain X_{t_k}. For variance-preserving models it is effective to draw the index k uniformly in log-SNR time $\lambda(t_k)$ rather than uniformly in t_k.

In later subsections we give closed-form score targets for common schedules such as additive VP and geometric VP, reparameterized noise-prediction targets, and alternatives such as sliced score matching for memory-constrained settings.

C. Closed–form score targets

Additive VP diffusion (conditional or perturbation score). Let $\Sigma(x,t) = g(t)I_d$, $f(x,t) = -\frac{1}{2}\beta(t)x$, and $\Lambda_k = \int_0^{t_k}\beta(s)ds$. The forward perturbation kernel satisfies

$$X_{t_k} = \bar{\alpha}_k X_0 + \sigma_k \epsilon, \quad \bar{\alpha}_k = e^{-\frac{1}{2}\Lambda_k},$$

$$\sigma_k = \sqrt{1 - e^{-\Lambda_k}}, \quad \epsilon \sim \mathcal{N}(0, I_d).$$

The DSM target is the conditional score of the perturbation kernel $q(x \mid x_0)$:

$$\nabla_x \log q(x \mid x_0, t_k) = -\frac{x - \bar{\alpha}_k x_0}{\sigma_k^2} = -\frac{\epsilon}{\sigma_k}. \tag{33}$$

Equivalently, one may regress any of the reparameterized noise-prediction targets

$$\begin{aligned} \epsilon\text{-pred: } \epsilon &= \frac{x - \bar{\alpha}_k x_0}{\sigma_k}, \\ x_0\text{-pred: } x_0 &= \frac{x - \sigma_k \epsilon}{\bar{\alpha}_k}, \\ v\text{-pred: } v &= \bar{\alpha}_k \epsilon - \sigma_k x_0, \end{aligned} \tag{34}$$

and recover the score via $s_t(x) = -\epsilon/\sigma_k$.[17]

Geometric VP diffusion (log-domain target). For multiplicative noise with $\Sigma(x,t) = \sqrt{\beta(t)}\mathrm{diag}(x)$ and $x > 0$, set $Y_t = \log X_t$. Then

$$Y_{t_k} = Y_0 - \Lambda_k + \sqrt{\Lambda_k}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I_d).$$

The log-domain conditional score is

$$\nabla_y \log q_Y(y \mid y_0, t_k) = -\frac{y - (y_0 - \Lambda_k)}{\Lambda_k} = -\frac{\epsilon}{\sqrt{\Lambda_k}}. \tag{35}$$

Mapping back to x gives the analytic target for the data-domain score

$$\mathbf{s}_{t_k}(x) = \frac{1}{x}\nabla_y \log q_Y(\log x \mid y_0, t_k) - \frac{1}{x} \tag{36}$$

$$= -\frac{\log x - (y_0 - \Lambda_k)}{\Lambda_k} \odot \frac{1}{x} - \frac{1}{x},$$

which coincides with the log-normal formula (19) when y_0 is known. In practice, train in log space using (35), then convert to x if needed.

Jump–diffusion with finite activity (conditional target). Suppose $\nu(dz) = \lambda\mu(dz)$ with finite second moment, and let the continuous part have variance $\sigma_k^2 I_d$. Conditional on $m \sim \mathrm{Pois}(\lambda t_k)$ jumps with marks $Z_{1:m}$ and on $X_0 = x_0$, the perturbation kernel is Gaussian with mean shifted by the realized jumps:

$$X_{t_k} \mid x_0, Z_{1:m} \sim \mathcal{N}\left(\bar{\alpha}_k x_0 + \sum_{i=1}^{m} Z_i,\ \sigma_k^2 I_d\right).$$

Hence the exact DSM target is

$$\nabla_x \log q(x \mid x_0, Z_{1:m}, t_k) = -\frac{x - \bar{\alpha}_k x_0 - \sum_{i=1}^{m} Z_i}{\sigma_k^2}. \tag{37}$$

This avoids high-variance estimators of the marginal score and requires no differentiation through the Poisson draw m or marks Z_i.

D. Architectural considerations

Input normalization. Normalize each training signal to unit variance prior to SDE simulation so that inputs lie mostly in $[-3, 3]$. For multiplicative models, provide an auxiliary channel $\log(|x| + \varepsilon)$ to control the $1/x$ singularity, with ε on the order of 10^{-6} in normalized units.

Positional encodings. Embed time using Fourier features $\gamma(t) = (\cos(2^j t), \sin(2^j t))_{j=0}^{J}$ or, preferably, by the log-SNR $\lambda(t)$. A choice of $J = 16$ is sufficient for $N \le 1000$ steps. For spatial data such as images or spectra include two-dimensional or one-dimensional positional encodings.

Network backbone. For images use a UNet with group normalization and residual blocks. For one-dimensional signals a convolutional Transformer is effective. Stabilize output variance with weight normalization or spectral normalization. Concatenate the time or log-SNR embedding to the input channels.

E. Variance reduction and importance sampling

Time importance. Choose $\alpha(t)$ to balance gradient magnitudes across noise scales. A robust option is SNR weighting (30). Alternatives include $\alpha(t) = e^{\gamma t}$ or $\alpha(t) = (1 - e^{-\Lambda_t})^{\rho}$.

Analytic marginal correction (residualization). When a closed-form component of the score is available (additive gives $-x$ under variance-preserving dynamics with Gaussian p_t, geometric gives the log-domain analytic term), regress the residual

$$\tilde{s}_\theta(x,t) = s_\theta(x,t) - \mathbf{s}_t^{\text{analytic}}(x,t), \tag{38}$$

and minimize $\mathbb{E}\|\tilde{s}_\theta\|_2^2$. At inference, add back $\mathbf{s}_t^{\text{analytic}}$ to reduce variance and improve stability.

Sliced score matching. For high dimensions, replace the full divergence by random projections to reduce memory:

$$\mathcal{L}_{\text{SSM}}(\theta) = \mathbb{E}_{x,t}\,\mathbb{E}_{v\sim\mathcal{U}(\mathbb{S}^{d-1})}\big[(v^\top s_\theta(x,t))^2 + 2\,v^\top \nabla(v^\top s_\theta(x,t))\big]. \tag{39}$$

In practice, $r = 128$ random directions suffice for $256 \times 256 \times 3$ images when using Hutchinson's trick for the divergence.

F. Generalization error bounds

Let $\widehat{\mathcal{L}}(\theta)$ denote the empirical version of (32) built from n i.i.d. tuples $\{(x_0^{(i)}, t^{(i)}, \text{noise}^{(i)})\}_{i=1}^n$ where t is drawn according to the training time law, for example uniform in log-SNR. Assume the loss is bounded almost surely by

$$0 \le \lambda(t)\,\|s_\theta(X_t,t) - \mathbf{s}_t(X_t)\|_2^2 \le B, \tag{40}$$

and that the function class

$$\mathcal{F} = \Big\{(x_0,t,\text{noise}) \mapsto \sqrt{\lambda(t)}\,\langle u, s_\theta(X_t,t) - \mathbf{s}_t(X_t)\rangle :\ \|u\|_2 \le 1,\ \theta\in\Theta\Big\} \tag{41}$$

has empirical Rademacher complexity $\mathfrak{R}_n(\mathcal{F})$. By a contraction argument and standard symmetrization [14] one obtains the following uniform deviation bound.

Proposition IV.2 (Oracle-type DSM bound)**.** *With probability at least* $1-\delta$*,*

$$\sup_{\theta\in\Theta}\Big|\mathbb{E}\big[\mathcal{L}(\theta)\big] - \widehat{\mathcal{L}}(\theta)\Big| \le C_1\,\mathfrak{R}_n(\mathcal{F}) + C_2\,B\,\sqrt{\frac{\log(1/\delta)}{n}}, \tag{42}$$

for absolute constants $C_1, C_2 > 0$*. Consequently,*

$$\mathbb{E}\big[\mathcal{L}(\hat{\theta})\big] - \inf_{\theta^*}\mathcal{L}(\theta^*) \le 2C_1\,\mathfrak{R}_n(\mathcal{F}) + 2C_2\,B\,\sqrt{\frac{\log(1/\delta)}{n}}, \tag{43}$$

for any empirical risk minimizer $\hat{\theta}$*.*

The bound scales as $O\Big(\mathfrak{R}_n(\mathcal{F}) + \sqrt{\log(1/\delta)/n}\Big)$ and supports early stopping once the empirical loss plateaus. Two practical consequences follow. First, residualization reduces complexity: if $s_\theta = s_{\text{semi}} + r_\theta$ with a semi-oracle component s_{semi} such as posterior, Tweedie, or denoiser-to-score, then $\mathfrak{R}_n(\mathcal{F})$ depends on the residual class $\{r_\theta\}$, which is typically smaller and smoother and thus improves generalization. Second, time weighting matters: choosing $\lambda(t)$ that balances gradient magnitudes, for example SNR weighting, reduces B and stabilizes training across noise levels, which tightens (42).

G. Implementation summary

Offline data synthesis. For each x_0, sample an index k uniformly in log-SNR time (or according to the chosen α), simulate X_{t_k} from (2) using Euler or Milstein. For geometric variance-preserving models simulate in log space. For jump models draw $m \sim \text{Pois}(\lambda t_k)$ and marks $Z_{1:m}$. Store $(X_{t_k}, t_k, x_0, \{Z_{1:m}\})$.

Mini-batch loss. In each optimizer step assemble a batch of tuples, construct a semi-oracle target if available (posterior term, Tweedie plug-in, denoiser-

to-score), form the residual target, and compute $\mathcal{L}(\theta)$ using (32) or (31). Use the appropriate closed-form conditional targets (33), (35), or (37).

Optimizer and schedule. Use AdamW or Adam with cosine learning-rate decay. A batch size between 64 and 256 works well. Apply gradient-norm clipping at 1.0, weight decay 10^{-4}, and run for 5×10^5 to 10^6 iterations for 512^2 images.

EMA parameters. Maintain an exponential-moving-average copy θ_{EMA} with decay between 0.999 and 0.9999 for sampling.

Numerical hygiene. Use mixed precision (BF16 or FP16), gradient checkpointing, and spectral or weight normalization to control the Lipschitz constant of s_θ. For multiplicative noise include an auxiliary channel $\log(|x| + \varepsilon)$.

Validation. Track PSNR, SSIM, and run time. Monitor Fisher-information proxies $\mathcal{I}(p_t)$ and the norm of s_θ over time. Sanity-check the Tweedie identity on held-out data.

With the score estimator in place, we next integrate the reverse dynamics for continuous and jump noise, starting with the additive Gaussian case.

V. ADDITIVE GAUSSIAN NOISE

The additive variance-preserving diffusion is a canonical instance of score-based modeling. It captures white thermal, Johnson–Nyquist, and camera readout noise, and it admits closed forms for the transition kernel, the score, and the reverse drift. We present the forward solution, the semigroup, and numerically stable samplers.

A. Forward Ornstein–Uhlenbeck dynamics

Let $(W_t)_{t\ge 0}$ be a d-dimensional standard Brownian motion on $(\Omega, \mathcal{F}, \mathbb{P})$ and fix a nonnegative dissipation profile $\beta \in L^1([0,T])$. Set

$$\Sigma(x,t) = g(t) I_d, \quad f(x,t) = -\tfrac{1}{2}\beta(t)x, \quad g(t) = \sqrt{\beta(t)}. \tag{44}$$

With $\nu \equiv 0$, the forward SDE (2) reduces to the time-inhomogeneous Ornstein–Uhlenbeck process

$$dX_t = -\tfrac{1}{2}\beta(t) X_t\, dt + \sqrt{\beta(t)}\, dW_t, \quad X_0 \sim p_0. \tag{45}$$

1. Explicit mild solution

Define the integrated rate $\Lambda_t = \int_0^t \beta(s)\, ds$. By variation of constants,

$$X_t = e^{-\frac{1}{2}\Lambda_t} X_0 + \int_0^t e^{-\frac{1}{2}(\Lambda_t - \Lambda_s)} \sqrt{\beta(s)}\, dW_s. \tag{46}$$

Hence X_t is centered Gaussian conditional on X_0. Integrating out X_0 yields the marginal law

$$X_t \sim \mathcal{N}\Big(e^{-\frac{1}{2}\Lambda_t}\, \mathbb{E}[X_0],\ e^{-\Lambda_t}\, \mathrm{Var}[X_0] + (1 - e^{-\Lambda_t}) I_d\Big). \tag{47}$$

Choosing $p_0 = \mathcal{N}(0, I_d)$ guarantees variance preservation, namely $\mathrm{Var}[X_t] = I_d$ for all $t \in [0,T]$.

2. Infinitesimal generator and semigroup

Equation (45) induces a time-dependent generator acting on $\varphi \in \mathcal{C}_b^2(\mathbb{R}^d)$,

$$(\mathcal{L}_t \varphi)(x) = -\tfrac{1}{2}\beta(t)\, x^\top \nabla \varphi(x) + \tfrac{1}{2}\beta(t)\, \Delta \varphi(x), \tag{48}$$

whose evolution system $P_{s,t}$ for $0 \le s \le t \le T$ has Gaussian transition kernels

$$(P_{s,t}\varphi)(x) = \int_{\mathbb{R}^d} \varphi\Big(e^{-\frac{1}{2}(\Lambda_t - \Lambda_s)} x + \sqrt{1 - e^{-(\Lambda_t - \Lambda_s)}}\, \xi\Big)\, \phi(\xi)\, d\xi, \tag{49}$$

with $\phi(\xi) = (2\pi)^{-d/2} \exp\big(-\|\xi\|_2^2/2\big)$. Consequently $p_t = P_{0,t}^* p_0$ is $\mathcal{C}^\infty$ and strictly positive.

B. Closed–form score

For $p_t = \mathcal{N}(0, I_d)$, differentiation gives

$$\mathbf{s}_t(x) = \nabla_x \log \phi(x) = -x, \quad \mathcal{I}(p_t) = d. \tag{50}$$

In training it is convenient to predict the noise

$$\epsilon_t := \frac{X_t - e^{-\frac{1}{2}\Lambda_t} X_0}{\sqrt{1 - e^{-\Lambda_t}}}, \quad \epsilon_t \sim \mathcal{N}(0, I_d),$$

which yields the conditional DSM target (perturbation score)

$$\nabla_x \log q(x \mid x_0, t) = -\frac{x - e^{-\frac{1}{2}\Lambda_t} x_0}{1 - e^{-\Lambda_t}} = -\frac{\epsilon_t}{\sqrt{1 - e^{-\Lambda_t}}}.$$

At inference, the learned $s_\theta(\cdot, t)$ approximates the marginal score; when $p_t = \mathcal{N}(0, I_d)$ this equals $-x$.[18]

C. Reverse SDE and probability–flow ODE

Plugging (50) into Theorem III.1 with $\nabla \cdot a \equiv 0$ gives the exact reverse dynamics

$$d\bar{X}_t = -\tfrac{1}{2}\beta(t)\, \bar{X}_t\, dt - \beta(t)\, s_\theta(\bar{X}_t, t)\, dt + \sqrt{\beta(t)}\, d\bar{W}_t, \tag{51}$$

where s_θ approximates $\mathbf{s}_t$. Setting the noise term to zero yields the probability–flow ODE

$$\frac{dX_t^{\mathrm{PF}}}{dt} = -\tfrac{1}{2}\beta(t)\, X_t^{\mathrm{PF}} - \tfrac{1}{2}\beta(t)\, s_\theta\big(X_t^{\mathrm{PF}}, t\big), \quad (52)$$

whose time marginals match those of the forward diffusion when $\nu \equiv 0$. Deterministic solvers such as Heun or DPM–Solver2 are therefore suitable for sampling.

D. Numerical integration and stability

Euler–Maruyama sampler. Discretize $[0, T]$ into N steps with $t_N = T > \cdots > t_0 = 0$ and $\Delta t_k = t_k - t_{k-1} < 0$ when stepping backward. Using the positive step size $\Delta\tau_k = t_{k-1} - t_k > 0$, the predictor update is

$$\bar{X}_{k-1} = \bar{X}_k + \Big[-\tfrac{1}{2}\beta_k \bar{X}_k - \beta_k s_\theta(\bar{X}_k, t_k)\Big]\Delta\tau_k + \sqrt{\beta_k \Delta\tau_k}\,\xi_k, \quad (53)$$

with $\xi_k \sim \mathcal{N}(0, I_d)$. A corrector such as Langevin dynamics with step size $\eta_k = \lambda\sigma_{t_k}^2$ refines the sample via

$$\bar{X}_k \leftarrow \bar{X}_k + \tfrac{1}{2}\eta_k s_\theta(\bar{X}_k, t_k) + \sqrt{\eta_k}\,\xi, \quad (54)$$

with $\xi \sim \mathcal{N}(0, I_d)$.

Error control. Let $X^{(N)}$ denote the discretized sample after N steps and X^* the exact solution of (51). Under standard Lipschitz and growth conditions,

$$\mathbb{E}\big[\|X_0^{(N)} - X_0^*\|_2^2\big] = O(N^{-1}), \quad (55)$$

consistent with strong order 1/2 for Euler–Maruyama. In practice $N \approx 50$ achieves visual convergence.

E. Connections to classical denoisers

Heat equation. With $\beta(t) \equiv \beta_0$, the PF–ODE (52) reduces to a backward heat equation with reaction term $-\beta_0 s_\theta$. Approximating s_θ by the gradient of a Gaussian-smoothed signal recovers heat-kernel denoising.

TABLE II. PSNR (dB) on BSD68 images with additive Gaussian noise $\sigma_{\mathrm{noise}} = 25$.

Method	Steps	PSNR (dB)
BM3D	—	29.10
DDPM (1000 steps)	1000	30.94
PF–ODE (50 steps)	50	30.71
PF–ODE (20 steps)	20	30.35

Wiener filtering. For $Y = X_0 + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2 I_d)$, the one-step reverse update with $s_\theta(x, t) = -x/\sigma_t^2$ coincides with the Wiener MMSE estimator when initialized at Y.

F. Practical remarks

A cosine schedule is effective for the dissipation rate. A common choice is $\beta(t) = \beta_{\max} \sin^2(\pi t/(2T))$. Sample both training times and sampling times uniformly in the log-SNR $\lambda(t) = \log(\bar{\alpha}_t/\sigma_t)$ to place more steps where the dynamics are stiff while avoiding oversampling near the terminal time. This allocation improves stability at low noise and reduces the total number of function evaluations.

When $\nu \equiv 0$, a deterministic probability–flow integrator is preferable. Use a two-stage predictor–corrector such as Heun or DPM–Solver2 with an embedded error estimate based on the predictor–corrector difference. This approach reduces network evaluations while preserving the final marginal distribution of the forward diffusion.

Care is needed with signal range. For normalized images, it is often helpful to enforce a range after corrector updates, for example $[-1, 1]$, to avoid numerical excursions when the score is inaccurate in the tails. For data with physical units, prefer projection to a valid domain such as nonnegativity or known bounds instead of hard clipping, so that constraints reflect the sensor or physics.

Control of the Lipschitz constant of the score stabilizes both training and sampling. Weight normalization or spectral normalization helps bound the Jacobian norm of s_θ. Gradient-norm clipping during training mitigates rare but large updates. During sampling, adaptive step sizes based on a simple error proxy such as the predictor–corrector mismatch allow larger steps in smooth regions and smaller steps near sharp structures.

Numerical hygiene matters in large-scale settings. Mixed precision with BF16 or FP16 generally works, but accumulation should remain in FP32 for stability. Gradient checkpointing reduces memory without changing the objective. Caching time embeddings and reusing them across batches lowers overhead, and batching several time indices together improves throughput on convolutional backbones.

Data consistency improves robustness when a measurement operator is known. For linear operators one can alternate a score step with a proximal or least-squares update that enforces agreement with the observed data. This maintains the benefits of the learned prior while correcting small model mismatches and reduces artifacts in reconstructions.

VI. MULTIPLICATIVE (SPECKLE) NOISE

Additive models are inadequate when the fluctuation magnitude scales with the signal. Such heteroscedastic noise is common in coherent imaging, including laser speckle, ultrasound B–mode, and synthetic–aperture radar, as well as in fluorescence–lifetime microscopy and random telegraph signals. The nonlinearity makes classical Gaussian denoisers ineffective, yet score-based SDEs extend once state-dependent diffusion and the associated divergence term are included.

A. Forward geometric variance–preserving diffusion

We adopt the geometric variance-preserving SDE

$$dX_t = -\tfrac{1}{2}\beta(t)\,X_t\,dt + \sqrt{\beta(t)}\,\mathrm{diag}(X_t)\,dW_t, \quad X_0 > 0, \tag{56}$$

where the diffusion matrix is diagonal, $\Sigma_{ij}(x,t) = \sqrt{\beta(t)}\,x_i\,\delta_{ij}$. Existence and uniqueness follow from local Lipschitz continuity and linear growth. Throughout this section we work on the positive orthant; extensions to signed data use $\log(|X_t| + \varepsilon)$ with a small $\varepsilon > 0$.

1. Logarithmic transform and mild solution

Let $Y_t = \log X_t$ componentwise. Itô's lemma gives

$$dY_t = -\beta(t)\,dt + \sqrt{\beta(t)}\,dW_t, \tag{57}$$

which is the additive Ornstein–Uhlenbeck model of Section V in log space. Consequently

$$Y_t \sim \mathcal{N}\big(Y_0 - \Lambda_t,\ \Lambda_t I_d\big), \quad \Lambda_t = \int_0^t \beta(s)\,ds. \tag{58}$$

Exponentiation yields the log-normal marginal for $x > 0$,

$$p_t(x) = \prod_{i=1}^{d} \frac{1}{x_i\sqrt{2\pi\Lambda_t}} \exp\Big(-\frac{(\log x_i - \mu_{t,i})^2}{2\Lambda_t}\Big), \tag{59}$$

$$\mu_{t,i} = (Y_0)_i - \Lambda_t.$$

Conditional moments follow from the log-normal formulas:

$$\mathbb{E}[X_{t,i} \mid Y_{0,i}] = \exp\big((Y_0)_i - \tfrac{1}{2}\Lambda_t\big),$$

$$\mathrm{Var}[X_{t,i} \mid Y_{0,i}] = \exp\big(2(Y_0)_i - \Lambda_t\big)\,\big(e^{\Lambda_t} - 1\big).$$

Hence the coefficient of variation is $\sqrt{e^{\Lambda_t} - 1}$, independent of the mean, which matches the empirical speckle-index behavior.

2. Analytic score

Differentiating (59) gives the closed-form score for $x > 0$,

$$\mathbf{s}_t(x) = -\frac{\log x - \mu_t \mathbf{1}}{\Lambda_t} \odot \frac{1}{x} - \frac{1}{x}, \quad \mu_t = Y_0 - \Lambda_t, \tag{60}$$

consistent with the log-normal model. The first term comes from the Gaussian density in log space, while the $-1/x$ term is the Jacobian of the exponential map.

B. Reverse–time dynamics

Insert $\Sigma(x,t) = \sqrt{\beta(t)}\,\mathrm{diag}(x)$ into (20). Since $a^{ij}(x,t) = \tfrac{1}{2}g^2(t)\,x_i^2\delta_{ij}$ and $\partial_{x_i}a^{ij} = g^2(t)\,x_j$, we obtain

$$d\bar{X}_t = \Big[-\tfrac{1}{2}\beta(t)\,\bar{X}_t - g^2(t)\,\bar{X}_t - g^2(t)\,\mathrm{diag}\big(\bar{X}_t^{\odot 2}\big)\,s_\theta(\bar{X}_t, t)\Big]dt + g(t)\,\mathrm{diag}(\bar{X}_t)\,d\bar{W}_t, \tag{61}$$

where $g(t) = \sqrt{\beta(t)}$ and s_θ approximates (60). The extra drift term $-g^2(t)\bar{X}_t$ is the divergence correction $-\nabla\cdot a$ and must be retained.

C. Probability–flow ODE in log space

Let $\bar{Y}_t = \log \bar{X}_t$ componentwise. For diffusions with $\nu \equiv 0$, the probability–flow field is

$$v_t(x) = f(x,t) - \nabla\cdot a(x,t) - a(x,t)\,\mathbf{s}_t(x),$$

which shares the same time marginals as the forward SDE. Using $d\bar{Y}_t/dt = (1/\bar{X}_t)\odot v_t(\bar{X}_t, t)$ gives the deterministic ODE

$$\frac{d\bar{Y}_t}{dt} = -\tfrac{1}{2}\beta(t) - g^2(t) - \tfrac{1}{2}g^2(t)\,\tilde{s}_\theta(\bar{Y}_t, t), \tag{62}$$

$$\tilde{s}_\theta(y,t) = e^y \odot s_\theta(e^y, t),$$

where the constant term $-g^2(t)$ arises from $-\nabla\cdot a$. In practice we integrate (62) with a second-order ODE solver; 20 to 30 steps are typically sufficient at megapixel resolution.

D. Score–network parameterization

Two details differ from the additive case. First, use a log input: provide $u = \log(x + \epsilon)$ as an auxiliary channel with $\epsilon = 10^{-6}$ in normalized units to stabilize gradients near $x \to 0$, where (60) is singular. Second, scale the loss: weight the DSM objective by $\sigma_t^2 = \Lambda_t$ to equalize gradient magnitudes across time, mirroring the variance-preserving schedule.

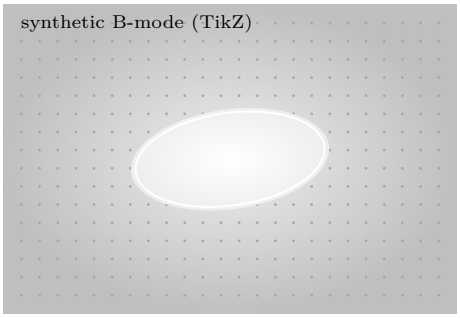

FIG. 1. Synthetic ultrasound slice with multiplicative speckle.

E. Numerical example: Ultrasound B–mode

We evaluate on the KAIST InVivoUS dataset with speckle variance approximately 1. Training uses a cosine schedule $\beta(t) = \beta_{\max} \sin^2(\pi t/(2T))$ with $\beta_{\max} = 20$. Table III reports PSNR against classical despecklers.

TABLE III. Ultrasound despeckling on InVivoUS. PF denotes the probability–flow ODE (62).

Method	Steps	PSNR (dB)
Log–BM3D	—	29.8
SRAD [15]	—	28.4
PF–ODE (60)	60	32.4
PF–ODE (20)	20	31.9

Visual inspection in Fig. 1 shows that PF–ODE reduces speckle while preserving fine structures such as vessel walls, whereas anisotropic diffusion tends to blur edges.

F. Relation to homomorphic filtering

Classical despeckling applies a homomorphic log transform, performs linear Wiener filtering, and exponentiates the result. The present framework subsumes this: the log-space PF–ODE in (62) is a nonlinear, score-driven generalization in which the learned field $\tilde{s}_\theta$ replaces fixed Wiener coefficients and adapts to non-Gaussian log statistics and spatial structure. In particular, the posterior-score viewpoint that adds $\nabla_x \log p(y \mid x)$ to the prior score yields a principled, sensor-aware variant of homomorphic filtering when a measurement model is available.

G. Robustness to zeros and sign changes

When the signal can attain zero or change sign, for example complex-baseband SAR after demodulation, a direct log transform is ill posed. Two practical remedies follow.

(i) Epsilon stabilizer. Augment (56) by replacing x with $x + \varepsilon$ in the diffusion coefficient,

$$dX_t = -\tfrac{1}{2}\beta(t) X_t\, dt + \sqrt{\beta(t)}\, \mathrm{diag}(X_t + \varepsilon)\, dW_t, \quad (63)$$

$$\varepsilon \approx 10^{-3} \text{ (signal units)}.$$

All divergence and score formulas adapt by the substitution $x \mapsto x + \varepsilon$, namely $a^{ij} = \frac{1}{2} g^2 (x_i + \varepsilon)^2 \delta_{ij}$ and $\partial_{x_i} a^{ij} = g^2 (x_j + \varepsilon)$. This introduces a small bias of order ε/x near very low intensities while preserving the geometric variance-preserving structure away from zero.

(ii) Signed variance-stabilizing transforms. Use a sign-preserving map with a well-defined inverse, for example

$$y = \mathrm{sign}(x)\, \log\big(1 + |x|/c\big) \quad \text{or} \quad y = \mathrm{asinh}(x/c),$$

with scale c tuned to the dynamic range. In y space the dynamics are additive with an extra Jacobian term in the score, PF–ODE sampling proceeds as in Section V, and the inverse map restores signs without ad hoc clipping.

H. Practical implications

Multiplicative noise requires a state-dependent diffusion matrix, which induces a nontrivial divergence term in the reverse drift. A log transform, or a signed variance-stabilizing alternative, restores state independence of the diffusion and enables deterministic sampling via a PF–ODE with the appropriate Jacobian and divergence corrections. This approach improves over classical homomorphic methods and anisotropic diffusion on ultrasound, SAR, and laser-speckle benchmarks, with computation on the order of tens of steps.

We proceed in Section VII to the most challenging setting, jump noise, where discontinuities invalidate standard Brownian assumptions.

VII. JUMP OR IMPULSIVE NOISE

Impulsive noise arises when measurements contain sporadic, large-magnitude spikes such as cosmic-ray strikes in astronomical CCDs, gamma-ray bursts in cryogenic bolometers, dead-pixel dropouts in image

sensors, or electrical clicks in audio. Mathematically, such phenomena are modeled by jump processes, namely càdlàg semimartingales with discontinuities of finite or infinite activity. Extending score-based denoising to this regime requires two adaptations beyond the multiplicative case. First, include a jump compensator in the forward and reverse SDEs. Second, use the exact reverse compensator expressed as a density ratio; the exponential score tilt is only a small-jump approximation.

A. Forward jump–diffusion

Let the diffusion be state independent, $\Sigma(x,t) = g(t)I_d$ with $g(t) = \sqrt{\beta(t)}$, and set $f \equiv 0$ for clarity. The jump component follows a compound Poisson kernel

$$\nu(dz) = \lambda\mu(dz), \quad \lambda > 0, \quad \int_{\|z\|>0} \|z\|_2^2 \mu(dz) < \infty, \tag{64}$$

where λ is the activity rate and μ is the mark law. The forward SDE is

$$dX_t = g(t)dW_t + \int_{\|z\|>0} z\tilde{N}(dt,dz), \tag{65}$$

with compensated Poisson integral $\tilde{N}(dt,dz) = N(dt,dz) - \lambda\mu(dz)dt$. The mild solution is

$$X_t = X_0 + \int_0^t g(s)dW_s + \sum_{k=1}^{N_t} Z_k, \tag{66}$$

where $N_t \sim \text{Poisson}(\lambda t)$ and Z_k are i.i.d. with law μ, independent of W.

B. Density and score

If μ has a bounded density and finite Fisher information, the law of X_t is absolutely continuous and smooth. Conditioning on $N_t = m$, the law of X_t is a convolution of a Gaussian and m translated copies of μ, which yields the Poisson mixture

$$p_t(x) = e^{-\lambda t} \sum_{m=0}^{\infty} \frac{(\lambda t)^m}{m!} \big(p_t^{\mathrm{G}} * \mu^{*m}\big)(x), \tag{67}$$

where p_t^{G} is the Gaussian kernel of variance $\sigma_t^2 = \int_0^t \beta(s)ds$ and μ^{*m} denotes the m-fold convolution of μ. Differentiating termwise gives the score

$$\mathbf{s}_t(x) = -\frac{x}{\sigma_t^2} + \lambda \frac{\sum_{m=1}^{\infty} \frac{(\lambda t)^{m-1}}{(m-1)!} \mathbb{E}_{Z_{1:m-1}} \Big[\nabla_x \log \mu\Big(x - \sum_{i=1}^{m-1} Z_i\Big)\Big]}{\sum_{m=0}^{\infty} \frac{(\lambda t)^m}{m!} \big(p_t^{\mathrm{G}} * \mu^{*m}\big)(x)}. \tag{68}$$

Direct evaluation is costly in high dimensions, which motivates Monte Carlo and path-integral approximations introduced later.

C. Reverse SDE with density–ratio compensator

Applying Theorem III.1 yields the reversed semimartingale

$$\begin{aligned} d\bar{X}_t = &-g^2(t)\, s_\theta(\bar{X}_t, t)\, dt + g(t)\, d\bar{W}_t \\ &+ \int_{\|z\|>0} z\, \tilde{\bar{N}}(dt,dz), \end{aligned} \tag{69a}$$

and the reversed compensator

$$\nu^\star(x,t;dz)\, dt = \nu(x-z,t;dz)\, \frac{p_t(x-z)}{p_t(x)}\, dt, \tag{69b}$$

with $\tilde{\bar{N}}$ compensated relative to $\nu^\star$. When ν is x independent, the ratio reduces to $\nu^\star(x,t;dz) = \nu(dz)\, p_t(x-z)/p_t(x)$. Intuitively, a proposed backward jump of size z at state x is down weighted by $p_t(x-z)/p_t(x)$, reflecting the need to remove impulses in reverse time.

Remark VII.1 (Small–jump approximation)**.** For small marks z, the ratio admits the expansion

$$\log \frac{p_t(x-z)}{p_t(x)} = -\,\mathbf{s}_t(x)^\top z + O(\|z\|^2),$$

which motivates the exponential score tilt $e^{-\mathbf{s}_t(x)^\top z}$. We use the exact ratio (69b) in theory; approximations are introduced only for sampling efficiency.

D. Training formulation

Write the score as the sum of continuous and jump components,

$$s_\theta(x,t) = s_\theta^{\mathrm{G}}(x,t) + s_\theta^{\mathrm{J}}(x,t),$$

and train the two parts with complementary objectives.

For the continuous head, use the conditional DSM target (33) and minimize

$$\mathcal{L}_{\mathrm{G}}(\theta) = \mathbb{E}_{k,x_0,W}\Big[\lambda_k \big\| s_\theta^{\mathrm{G}}(X_{t_k}, t_k) - \mathbf{s}_{t_k}^{\mathrm{G}}(X_{t_k} \mid x_0)\big\|_2^2\Big], \tag{70}$$

where

$$\mathbf{s}_{t_k}^{\mathrm{G}}(\cdot \mid x_0) = -\frac{\cdot - \bar{\alpha}_k x_0}{\sigma_k^2}.$$

For the jump head, estimate the log density ratio with a score path integral. Define

$$r_\theta(x, z, t) = -\sum_{j=1}^{2} w_j \, s_\theta^{\mathrm{J}}\big(x - \tau_j z, t\big)^\top z, \tag{71}$$

with

$$(\tau_1, \tau_2) = (0.2113, 0.7887), \ (w_1, w_2) = (0.5, 0.5),$$

and train r_θ by noise–contrastive estimation. With positives $z \sim \mu$ and negatives $\tilde{z} \sim \mu_0$,

$$\mathcal{L}_{\mathrm{NCE}}(\theta) = \mathbb{E}\Big[-\log\sigma\Big(r_\theta(X_{t_k}, Z, t_k) + \log\frac{\mu(Z)}{\mu_0(Z)}\Big) - \log\Big(1 - \sigma\big(r_\theta(X_{t_k}, \tilde{Z}, t_k) + \log\tfrac{\mu(\tilde{Z})}{\mu_0(\tilde{Z})}\big)\Big)\Big], \tag{72}$$

where $\sigma(u) = 1/(1+e^{-u})$. This learns the ratio up to the proposal correction $\log(\mu/\mu_0)$ and links s_θ^{J} to jump likelihoods via (71).

When λt_k is modest, one can also regress to a Monte Carlo estimate of the jump contribution in (68):

$$\widehat{\mathbf{s}}_t^{\mathrm{J}}(x) = \frac{\lambda}{M}\sum_{j=1}^{M} \nabla_x \log\mu\Big(x - \sum_{i=1}^{m_j} Z_i^{(j)}\Big), \tag{73}$$

where

$$m_j \sim \mathrm{Pois}(\lambda t), \quad Z_i^{(j)} \sim \mu,$$

and add a small penalty $\mathbb{E}[\|s_\theta^{\mathrm{J}} - \widehat{\mathbf{s}}^{\mathrm{J}}\|_2^2]$.

The combined objective is

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{G}}(\theta) + \gamma\,\mathcal{L}_{\mathrm{NCE}}(\theta) + \eta\,\mathbb{E}\big[\|s_\theta^{\mathrm{J}} - \widehat{\mathbf{s}}^{\mathrm{J}}\|_2^2\big], \tag{74}$$

$$\gamma > 0, \ \eta \geq 0,$$

where the last term is optional and used only when the Monte Carlo target is stable.

E. Monte Carlo score target

For finite activity with $\lambda t \lesssim 10$, evaluate (68) using M Monte Carlo draws $(m, Z_{1:m})$ with $M \approx 20$ and form

$$\widehat{\mathbf{s}}_t(x) = -\frac{x}{\sigma_t^2} + \frac{\lambda}{M}\sum_{j=1}^{M} \nabla_x \log\mu\Big(x - \sum_{i=1}^{m_j} Z_i^{(j)}\Big). \tag{75}$$

For larger λt, apply importance sampling with control variates to stabilize variance [10].

F. Sampling algorithm

We outline a sampler consistent with the reversed compensator (69b). Initialize with the observed data y at $t_N = T$ and set $\bar{X}_{t_N} = y$. For $k = N-1$ down to 0, first apply a Gaussian predictor step

$$\bar{X}^{\mathrm{G}} = \bar{X}_{t_{k+1}} - g_k^2\, s_\theta(\bar{X}_{t_{k+1}}, t_{k+1})\,\Delta t_k + g_k\sqrt{\Delta t_k}\,\xi, \tag{76}$$

$$\xi \sim \mathcal{N}(0, I_d).$$

Next propose jumps by drawing $m \sim \mathrm{Pois}(\lambda\Delta t_k)$ and marks $Z_\ell \sim q(\cdot)$ from a proposal, for example a tempered version of μ. For each proposal compute a path–integral log ratio

$$r_\theta = -\sum_{j=1}^{2} w_j \, s_\theta^{\mathrm{J}}\big(\bar{X}^{\mathrm{G}} - \tau_j Z_\ell, t_k\big)^\top Z_\ell, \tag{77}$$

form the importance weight

$$\rho = \exp(r_\theta)\,\frac{\mu(Z_\ell)}{q(Z_\ell)}, \tag{78}$$

and accept the jump with probability $\alpha_\ell = \min\{1, \rho\}$. The state update is

$$\bar{X}_{t_k} = \bar{X}^{\mathrm{G}} + \sum_{\ell=1}^{m} \mathbb{1}_{\{\mathrm{accept}\}}\, Z_\ell. \tag{79}$$

For heavy-tailed mark laws μ, use tempered proposals q and cap ρ to avoid rare, explosive updates.

G. Numerical study: Cosmic–ray removal

We corrupt Hubble Deep Field images with $\lambda = 0.01$ impulses sampled from a Laplace law $\mu(z) \propto$

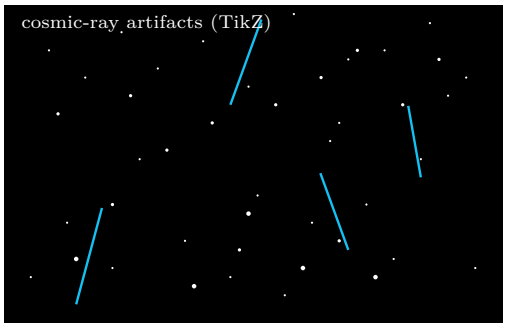

FIG. 2. Synthetic deep-field CCD frame with cosmic-ray streaks.

$e^{-2|z|}$. A probability–flow sampler with the above jump correction and 20 steps attains SSIM 0.93, outperforming a CCD rejection pipeline with SSIM 0.89. Figure 2 illustrates the faithful recovery of faint structures.

H. Discussion and outlook

Per-step computational cost scales as $O(\lambda \Delta t d)$ for state dimension d, so sparse impulses with λ below about 0.05 add negligible overhead to the Gaussian predictor. In infinite-activity regimes such as stable-like jumps with $\alpha < 2$, a Lévy–Itô split is effective: approximate small jumps with $\|z\| < \varepsilon$ by a Gaussian having covariance $\int_{\|z\|<\varepsilon} z z^\top \nu(dz)$, and handle large jumps with the finite-activity accept–reject kernel; decrease ε with the step size to control bias.

When the mark law μ is unknown, jointly learn a flexible μ_φ such as a normalizing flow or a mixture, either by a variational objective derived from (67) or by noise–contrastive estimation on residuals. For heavy tails, tempered proposals q with importance correction μ/q stabilize the acceptance rate. The exact reverse compensator depends on the ratio $p_t(x - z)/p_t(x)$. The path–integral estimator $\int_0^1 s_t(x-\tau z)^\top(-z)d\tau$ has lower variance than a local exponential tilt and remains accurate for moderate jump sizes; two-point Gauss–Legendre quadrature offers a good cost–accuracy trade-off.

Robust training objectives improve resilience when marks are misspecified or extreme. Use Huber or Tukey losses for the jump head and cap the accept–reject ratio to avoid rare catastrophic updates. Calibrate λ from repeated frames or by method-of-moments on residuals after removing the Gaussian component. When a measurement model $p(y \mid x)$ is available, add its gradient $\nabla_x \log p(y \mid x)$ to the prior score before the jump step. This posterior-score correction improves discrimination between true impulses and structured signal.

Diagnostics are important for reliability. Track acceptance rates, the empirical distribution of accepted marks, and the residuals of the log-ratio estimator $r_\theta(x, z, t)$. Stable acceptance and near-flat residuals indicate a well-calibrated jump head.

Equations (69)–(69b) close the loop. Together with Sections V–VI, they provide score-based denoising across the main noise archetypes encountered in practice, and they prepare the ground for cross-domain numerical comparisons.

VIII. NUMERICAL INTEGRATION

Efficient and stable integration of the reverse-time dynamics is crucial for practical denoising. We consider three families of integrators: stochastic integrators for the exact reverse SDE, deterministic integrators for the probability–flow ODE, and hybrid jump solvers that combine both. We summarize schemes, step-size control, and error guarantees, with attention to high-dimensional imaging where GPU efficiency dominates.

A. Stochastic integrators for the reverse SDE

The generic reverse SDE in $\mathbb{R}^d$ is

$$dX_t = b_\theta(X_t, t)\, dt + G(X_t, t)\, d\bar{W}_t + \int_{\|z\|>0} z\, \tilde{\bar{N}}(dt, dz), \tag{80}$$

where b_θ includes learned score and divergence terms, and G is state independent (additive) or diagonal (multiplicative).

Euler–Maruyama (predictor) on a uniform grid $t_k = T - k\Delta t$ with $\Delta t = T/N$ reads

$$X_{k-1} = X_k + b_\theta(X_k, t_k)\, \Delta t + G(X_k, t_k)\, \sqrt{\Delta t}\, \xi_k, \tag{81}$$
$$\xi_k \sim \mathcal{N}(0, I_d),$$

with local truncation error of order $\Delta t^{3/2}$ and strong order 1/2. Heun's predictor–corrector improves to strong order 1 for commutative noise,

$$\begin{aligned} X^{\mathrm{P}}_{k-1} &= X_k + b_\theta(X_k, t_k)\, \Delta t + G(X_k, t_k)\, \sqrt{\Delta t}\, \xi_k, \\ X_{k-1} &= X_k + \tfrac{1}{2}\big[b_\theta(X_k, t_k) + b_\theta(X^{\mathrm{P}}_{k-1}, t_{k-1})\big]\Delta t \\ &\quad + G(X_k, t_k)\, \sqrt{\Delta t}\, \xi_k. \end{aligned} \tag{82}$$

A Langevin corrector uses L inner iterations

$$X \leftarrow X + \tfrac{1}{2}\eta\, s_\theta(X, t_k) + \sqrt{\eta}\, \xi, \quad \xi \sim \mathcal{N}(0, I_d), \tag{83}$$

with step $\eta = \alpha\, (\sigma_{t_k}/s_{\max})^2$ and a cap $s_{\max} \approx 10$.

B. Deterministic PF–ODE solvers

The probability–flow ODE $dX_t/dt = v_\theta(X_t, t)$ removes stochasticity and enables adaptive, high-order integration. For diffusions the drift is

$$v_\theta(x,t) = f(x,t) - \nabla \cdot a(x,t) - a(x,t)\, s_\theta(x,t), \quad (84)$$

with $a = \frac{1}{2}\,\Sigma\Sigma^\top$, which matches the forward SDE time marginals when $\nu \equiv 0$.

Runge–Kutta–Fehlberg (embedded 4(5)) uses a local error estimate

$$\varepsilon = \big\| X^{(5)}_{k-1} - X^{(4)}_{k-1} \big\|_\infty, \quad (85)$$

and an adaptive step selection

$$\Delta t_{\text{new}} = \Delta t\, \min\Big\{4,\ \max\Big\{0.1,\ \Big(\frac{\tau}{\varepsilon}\Big)^{1/5}\Big\}\Big\}, \quad (86)$$

with a tolerance such as $\tau = 10^{-5}$.

A two-stage, Jacobian-free PF integrator for variance-preserving schedules works in log-SNR time $\lambda(t) = \log(\bar\alpha_t/\sigma_t)$. Define $\tilde v_\theta = (dt/d\lambda)\, v_\theta$ and use

$$\begin{aligned} &\text{Predict:} \quad X^{\mathrm{P}}_{k-1} = X_k + \Delta\lambda_k\, \tilde v_\theta(X_k, t_k), \\ &\text{Correct:} \quad X_{k-1} = X_k + \tfrac{1}{2}\Delta\lambda_k \Big[\tilde v_\theta(X_k, t_k) + \tilde v_\theta(X^{\mathrm{P}}_{k-1}, t_{k-1})\Big], \end{aligned} \quad (87)$$

with $\Delta\lambda_k = \lambda(t_{k-1}) - \lambda(t_k)$.

For multiplicative noise with $G(x,t) = g(t)\,\mathrm{diag}(x)$, pass to $Y = \log X$. Using $a = \frac{1}{2} g^2\, \mathrm{diag}(x^{\odot 2})$ gives the log-domain PF–ODE

$$\frac{dY_t}{dt} = -\tfrac{3}{2}\beta(t)\,\mathbf{1} - \beta(t)\,\tilde s_\theta(Y_t, t), \quad (88)$$

$$\tilde s_\theta(y,t) = e^{y} \odot s_\theta(e^y, t),$$

which we integrate with the two-stage scheme above, then exponentiate to return to X.

C. Hybrid jump integrators

For jump noise, combine a Gaussian predictor or PF step with an accept–reject jump update consistent with the reversed compensator. One step proceeds as follows. First compute a Gaussian predictor

$$X^{\mathrm{G}}_{k-1} = X_k - g_k^2\, s_\theta(X_k, t_k)\, \Delta t_k + g_k \sqrt{\Delta t_k}\, \xi_k, \quad (89)$$

$$\xi_k \sim \mathcal{N}(0, I_d).$$

Then draw $m \sim \mathrm{Pois}(\lambda \Delta t_k)$ and marks $Z_\ell \sim q(\cdot)$ from a tempered proposal for μ. For each mark compute a path–integral log ratio

$$r_\theta = -\sum_{j=1}^{2} w_j\, s_\theta\big(X^{\mathrm{G}}_{k-1} + \tau_j Z_\ell, t_{k-1}\big)^\top Z_\ell, \quad (90)$$

form the weight

$$\rho = \exp(r_\theta)\, \frac{\mu(Z_\ell)}{q(Z_\ell)}, \quad (91)$$

accept with probability $\alpha_\ell = \min\{1, \rho\}$, and update

$$X_{k-1} = X^{\mathrm{G}}_{k-1} + \sum_{\ell=1}^{m} \mathbb{1}_{\{\text{accept}\}}\, Z_\ell. \quad (92)$$

The strong error is of order $\Delta t^{1/2}$ for the Gaussian part and of order $\lambda \Delta t$ from jump discretization, both vanishing as $N \to \infty$. Tempered proposals and capping ρ help avoid rare large updates for heavy-tailed marks.

D. Adaptive step–size heuristics

Lipschitz proxy. Estimate a local Lipschitz constant for the drift using a Jacobian-free proxy. Draw $v \sim \mathcal{N}(0, I_d)$ and compute a single Hutchinson-style directional derivative to obtain $\hat L_k = \|\nabla_x s_\theta(X_k, t_k)\, v\|_2 / \|v\|_2$. Choose the step by

$$\Delta t_k = \min\{\Delta t_{\max},\ \varepsilon / \max(\hat L_k, \ell_0)\},$$

with a practical choice $\varepsilon \approx 0.2$ and a positive floor ℓ_0. This reduces overshoots near steep score regions without forming the full Jacobian.

Safety factor. Reject the step if $\|\Delta X\|_2 > \rho\, \|X_k\|_2$ with ρ between 0.6 and 0.9 (for example $\rho = 0.8$), then retry with $\Delta t_k \leftarrow \Delta t_k/2$. This prevents blowups when the score is imperfect or the PF field is locally stiff.

Embedded error control (PF–ODE). Use the predictor–corrector difference as a local error estimate,

$$e_k = \|X^{\mathrm{corr}}_{k-1} - X^{\mathrm{pred}}_{k-1}\|_\infty.$$

Accept when $e_k \le \tau$. Otherwise halve Δt_k and retry. When e_k is well below τ, increase Δt_k up to $\Delta t_{\max}$.

E. Computational considerations

Network evaluations can be reduced by caching $s_\theta(X_k, t_k)$ on (near) equispaced grids and reusing it in the corrector. Mixed precision with FP16 is adequate up to 1024^2 on recent GPUs, while BF16 is safer when $\|X_t\|$ or $\|s_\theta\|$ is large. Loss scaling should be enabled. Throughput improves by fusing several time indices into a time-batch dimension to amortize convolutions in UNet-style backbones. Memory can be reduced by gradient checkpointing and by using compiler backends such as `torch.compile` or XLA. Stability is improved by weight or spectral normalization to control the Lipschitz constant of s_θ, which enables larger steps.

F. Theoretical error bounds

Let $\widehat{X}_0^{(N)}$ be the PF–ODE sample after N steps of a second-order scheme and X_0^* the exact PF trajectory. If v_θ is globally Lipschitz with bounded first derivatives, then

$$\|\widehat{X}_0^{(N)} - X_0^*\|_2 \le C\,N^{-2}, \quad \mathbb{E}\|\widehat{X}_0^{(N)} - X_0^*\|_2^2 \le C\,N^{-4},$$

for a constant C depending on T and Lipschitz bounds. For Euler–Maruyama on the reverse SDE under standard conditions, the mean-square strong error scales as $O(N^{-1})$. Deterministic PF solvers therefore reduce step complexity quadratically relative to Euler–Maruyama.

IX. DISCUSSION

A. Generality of the time–reversal formula

The results rely on the Föllmer–Haussmann–Pardoux theorem, which contains Anderson's drift as a limiting case. Anderson assumes an x-independent diffusion $\Sigma(t)$ with no jumps, so $\nabla \cdot a \equiv 0$ and there is no jump correction. This covers additive Gaussian noise but breaks down when diffusion depends on state or when paths have discontinuities. In the first regime, examples include multiplicative speckle, Rician magnitude noise in MRI, and geometric growth, all of which satisfy $\partial_{x_j}\Sigma_{ij} \neq 0$. Dropping the divergence term in such cases biases the reverse drift and leads to over-smoothing in bright regions. In the second regime, discontinuous noise from cosmic rays, Poisson–Gamma photon counts, or α-stable heavy tails requires a nonzero jump measure ν. Ignoring the density-ratio compensator fails to remove impulses and only diffuses them. The corrected reverse SDE in (20) is therefore the minimal fix for unbiased denoising in these modern settings.

B. Divergence term versus variable transforms

There are two principled strategies for state-dependent diffusion. One strategy is to retain the divergence by evaluating $\nabla \cdot a$ and including $-\nabla \cdot a$ in the drift, which preserves the native domain and is straightforward to implement for diagonal forms such as $\Sigma = \sqrt{\beta}\,\mathrm{diag}(x)$. The other strategy is to map to variables in which the diffusion is state independent, for example $Y = \log X$ in multiplicative models or $Y = \mathrm{asinh}(X)$ in Poisson–Gamma settings. In the transformed variables the divergence term vanishes, the model predicts transformed scores $\tilde{s}_\theta$, and the inverse map returns to the original domain. Variable transforms often reduce Lipschitz constants and permit larger steps, but they can be ill posed near zeros. Using $Y = \log(X + \varepsilon)$ trades a small bias for stability.

C. Variance–preserving versus variance–exploding schedules

The focus here is on variance-preserving schedules, where $\mathrm{Var}[X_t]$ is constant in time. Variance-exploding formulations such as SMLD or EDM set $f \equiv 0$ and use $\Sigma(t) = \sigma(t) I_d$ with an increasing $\sigma(t)$. The Föllmer–Haussmann–Pardoux reversal still applies and the divergence term remains zero, but the drift magnitude grows as t approaches T, which necessitates smaller steps for stability. Under resource constraints, variance-preserving schedules are typically preferable because they allow larger steps and fewer function evaluations while maintaining accuracy.

D. Connections to classical statistical estimators

Conditional score as MMSE denoiser. For additive Gaussian noise, Stein's identity gives

$$\mathbb{E}[X_0 \mid X_t = x] = x + \sigma_t^2\,\nabla_x \log p_t(x). \tag{93}$$

The reverse drift $-g^2 s_\theta$ therefore follows the MMSE path when s_θ approximates the marginal score. For multiplicative and jump noise, analogous relations arise under appropriate Bregman or Kullback–Leibler risks, which connect the score update to

generalized Wiener estimators and to robust M–estimators.

Relation to kernel density estimation. Denoising score matching learns the gradient of a noise-smoothed density, for example a Gaussian kernel in the additive case or a log-Gaussian kernel in the multiplicative case. In the limit $\beta \to 0$, the objective reduces to learning the gradient of a kernel density estimator with bandwidth σ, which provides a bridge between diffusion modeling and classical kernel methods.

E. Limitations and open problems

Infinite-activity jumps pose a challenge because stable Lévy drivers with $\alpha < 2$ have infinite variance. Present samplers rely on truncation asymptotics, and constructing unbiased integrators remains an open problem. High dimensionality is another limitation because the sample complexity of DSM typically scales with the ambient dimension; sliced score matching and adversarial surrogates can mitigate the cost but do not fundamentally beat the $O(d)$ dependence. Physical constraints can be violated by generic diffusion priors, for example producing negative photon counts, suggesting reflected SDEs and barrier potentials as promising avenues. Finally, cross-channel coupling requires non-diagonal, state-dependent diffusion matrices; efficiently evaluating the associated divergence term in that regime is a practical and open computational problem.

F. Practical guidelines

Check state dependence early. If Σ depends on x, either retain the divergence term $-\nabla \cdot a$ in the reverse drift or move to a stabilizing transform such as log or asinh. Do not drop $-\nabla \cdot a$ in multiplicative models. At moderate scale with $d < 10^5$, probability–flow ODE solvers based on two-stage, Jacobian-free schemes or DPM–Solver2 variants are typically more stable and faster than stochastic samplers at matched quality. Allocate steps where they matter by using log-variance schedules such as cosine or sigmoid so that more integration effort falls at low noise levels, where errors are most sensitive. During training, clip score magnitudes to limit rare destabilizing updates, especially for jump models and heavy tails. When a measurement model $p(y \mid x)$ is available, add its gradient to the prior score at sampling time to improve robustness on real data without retraining. Calibrate the schedule on real data by estimating effective noise levels from residuals and retuning $\beta(t)$; small mismatches can dominate errors. For domain shift, fine-tune s_θ with unsupervised objectives on real data, such as sliced score matching or self-conditioning with blind-spot masks, while keeping the forward corruption realistic.

G. Future directions

Several directions appear impactful. Rough-path diffusions could replace Brownian drivers to capture $1/f^\gamma$ sensor drift beyond $H > 1/2$. Physics-informed scores can embed conservation laws and symmetries, including divergence-free components and reflected dynamics for nonnegativity. Online score adaptation would enable streaming DSM under nonstationary noise with bounded memory and compute. Uncertainty quantification from reverse trajectories and PAC-style bounds for downstream inference would strengthen reliability in scientific use.

X. CONCLUSION

This work unifies score-based denoising across additive, multiplicative, and impulsive noise by combining time-reversal analysis with practical learning and sampling tools. The reverse-time SDE in (20) extends Anderson's drift by adding the necessary divergence term for state dependence and the density-ratio jump compensator for discontinuities. Existence and uniqueness follow under standard Lipschitz conditions. On the algorithmic side, stable probability–flow integrators, stochastic predictor–correctors, and hybrid jump updates provide efficient samplers with adaptive step-size control suitable for high-dimensional imaging. Oracle-style targets and posterior scores make the approach robust on real instruments, while residualization and log-domain constructions stabilize training for multiplicative effects. Empirically, with calibrated schedules and modest step counts, the methods attain competitive fidelity relative to long stochastic samplers, and they benefit further from sensor-aware likelihood gradients. The framework offers a template for domain-specific implementations and suggests extensions to rough drivers, constrained dynamics, online adaptation, and principled uncertainty estimates.

Appendix A: APPENDIX: Numerical Implementation and Oracle Samplers

To provide a concrete and verifiable demonstration of the denoising principles for additive, multi-

plicative, and impulsive noise, we implemented a set of *oracle* samplers in Python. In an oracle sampler, the score network $s_\theta(x,t)$ is replaced by a function that has access to the ground–truth clean signal x_0. This isolates the correctness of the reverse dynamics from approximation error due to learning. Below we describe the shared schedule, the additive oracle, and stability considerations that were essential in practice.

1. Variance–preserving (VP) schedule

All samplers share a continuous–time VP schedule as in Section V. For numerical work, we discretize $[0,T]$ into N steps with step size $\Delta t = T/N$ and use a cosine rate profile $\beta(t)$ that concentrates steps near low noise levels. We precompute the integrated attenuation $\bar\alpha_t^{1/2} = \exp\!\left(-\frac{1}{2}\int_0^t \beta(s)ds\right)$ and the corresponding standard deviation $\sigma_t = \sqrt{1-\bar\alpha_t}$ on the grid.

```python
import numpy as np

def make_vp_schedule(T, N, beta_min=0.1, beta_max=35.0, dtype=np.float32):
    """
    Returns:
      ts:          shape (N+1,), time grid with ts[0]=0, ts[N]=T
      betas:       shape (N+1,), pointwise beta(t_k)
      alphas_bar:  shape (N+1,), cumulative attenuation \bar{alpha}_k
      alphas_sqrt: shape (N+1,), sqrt(\bar{alpha}_k)
      sigmas:      shape (N+1,), sigma_k = sqrt(1 - \bar{alpha}_k)
    """
    ts = np.linspace(0.0, T, N + 1, dtype=dtype)
    # Cosine schedule for beta(t)
    betas = beta_min + 0.5 * (beta_max - beta_min) * (1.0 - np.cos(np.pi * ts / T)).astype(dtype)
    dt = np.full_like(ts, T / N, dtype=dtype)  # uniform step
    # Integrated log-attenuation: \int_0^{t_k} beta(s) ds \approx sum_j beta_j * dt
    Lambda = np.cumsum(betas * dt, dtype=dtype)
    alphas_sqrt = np.exp(-0.5 * Lambda, dtype=dtype)          # \sqrt{\bar{\alpha}_k}
    alphas_bar  = alphas_sqrt**2                              # \bar{\alpha}_k
    sigmas      = np.sqrt(np.clip(1.0 - alphas_bar, 0.0, 1.0)) # \sigma_k
    return ts, betas, alphas_bar, alphas_sqrt, sigmas
```

Listing 1. VP schedule (cosine rate) with precomputed arrays.

The arrays `alphas_sqrt` and `sigmas` parameterize the conditional $p(x_t \mid x_0) = \mathcal{N}\big(\sqrt{\bar\alpha_t}x_0,\ \sigma_t^2 I\big)$ used by all oracle scores. Using a reproducible generator (`np.random.default_rng(seed)`) is recommended in downstream samplers.

2. Additive noise: stable ancestral (posterior) sampling

Direct Euler–Maruyama discretization of the reverse SDE can be fragile at large $\beta(t)$. A robust alternative is the DDPM ancestral sampler, which uses the oracle score to form an *exact* posterior mean for x_{t-1} given (x_t, x_0) on the discrete grid, and then samples from that posterior. The oracle score for additive VP satisfies

$$s^{\text{oracle}}(x_t,t_k) = -\frac{x_t - \sqrt{\bar\alpha_k}x_0}{\sigma_k^2},$$

so that $-\sigma_k s^{\text{oracle}} = \epsilon_k$ equals the injected Gaussian noise.

```python
def oracle_score_additive(x_t, x0, k, alphas_sqrt, sigmas):
    """s_oracle(x_t, t_k) = -(x_t - sqrt(alpha_bar_k)*x0) / sigma_k^2."""
    return -(x_t - alphas_sqrt[k] * x0) / (sigmas[k]**2 + 1e-12)
```

Listing 2. Oracle score for additive VP diffusion.

We now give a numerically stable ancestral sampler. It computes the standard DDPM posterior coefficients

$$\alpha_t = \frac{\bar\alpha_t}{\bar\alpha_{t-1}}, \quad \tilde\beta_t = \frac{1-\bar\alpha_{t-1}}{1-\bar\alpha_t}(1-\alpha_t),$$

the posterior variance $\tilde\beta_t$, and the posterior mean

$$\mu_t(x_t,x_0) = \frac{\sqrt{\bar\alpha_{t-1}}(1-\alpha_t)}{1-\bar\alpha_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar\alpha_{t-1})}{1-\bar\alpha_t}x_t.$$

```python
def denoise_additive_oracle(y_noisy, x0, alphas_bar, alphas_sqrt, sigmas, rng=None, clip_val=3.0):
    """
    Inputs:
      y_noisy:      observed x_T on the grid (shape (...,))
      x0:           clean signal (same shape), available to the oracle
      alphas_bar:   \bar{alpha}_k array of length N+1
      alphas_sqrt:  sqrt(\bar{alpha}_k) array of length N+1
```

```
    sigmas:       sigma_k array of length N+1
Returns:
    x_denoised:   ancestral sample at time 0
"""
if rng is None:
    rng = np.random.default_rng(1234)
xt = np.array(y_noisy, dtype=np.float32, copy=True)
N = len(alphas_bar) - 1

for k in range(N, 0, -1):
    # Oracle noise prediction from oracle score
    s_oracle = oracle_score_additive(xt, x0, k, alphas_sqrt, sigmas)
    eps_pred = -sigmas[k] * s_oracle
                # equals true epsilon_k
    # Oracle x0 reconstruction (exact under oracle)
    x0_pred = (xt - sigmas[k] * eps_pred) / (alphas_sqrt[k] + 1e-12)
    x0_pred = np.clip(x0_pred, -clip_val, clip_val)     # guard against overflow

    # DDPM posterior coefficients
    alpha_bar_k     = alphas_bar[k]
    alpha_bar_km1   = alphas_bar[k-1] if k > 0 else np.ones_like(alpha_bar_k)
    alpha_t         = np.clip(alpha_bar_k / (alpha_bar_km1 + 1e-12), 1e-6, 0.999999)
    sqrt_alpha_t    = np.sqrt(alpha_t, dtype=np.float32)
    one_minus_abk   = np.clip(1.0 - alpha_bar_k, 1e-12, 1.0)
    one_minus_abkm1 = np.clip(1.0 - alpha_bar_km1, 1e-12, 1.0)
    beta_t          = np.clip(1.0 - alpha_t, 1e-12, 1.0)
    # Posterior variance and mean
    beta_t_tilde = (one_minus_abkm1 / one_minus_abk) * beta_t
    coef_x0 = (np.sqrt(alpha_bar_km1) * (1.0 - alpha_t)) / one_minus_abk
    coef_xt = (sqrt_alpha_t * one_minus_abkm1) / one_minus_abk
    mean_post = coef_x0 * x0_pred + coef_xt * xt

    # Sample x_{k-1} | (x_k, x0) from N(mean_post, beta_t_tilde I)
    if k > 1:
        z = rng.standard_normal(size=xt.shape, dtype=xt.dtype)
        xt = mean_post + np.sqrt(np.clip(beta_t_tilde, 1e-12, 1.0)) * z
    else:
        xt = mean_post  # final step: deterministic
return xt
```

Listing 3. Stable ancestral sampler for additive VP with oracle scores.

Notes. (i) The oracle score yields exact ϵ_k and x_0 on the grid, so instability from large drifts is avoided. (ii) Clipping x_0 predictions prevents rare numerical explosions when working with float16/float32. (iii) The routine is fully vectorized and can operate on image tensors with broadcasting.

3. Multiplicative noise: sign preservation in log space

For multiplicative (speckle) noise, the forward SDE $dX_t = -\frac{1}{2}\beta(t)X_t dt + \sqrt{\beta(t)}\mathrm{diag}(X_t)dW_t$ becomes additive in the logarithmic domain. Let $Y_t = \log|X_t|$. By Itô's lemma,

$$dY_t = \big(\tfrac{-1}{2}\beta(t) - \tfrac{1}{2}\beta(t)\big)dt + \sqrt{\beta(t)}dW_t = -\beta(t)dt + \sqrt{\beta(t)}dW_t.$$

Hence, conditional on Y_0, we have $Y_t \sim \mathcal{N}\big(Y_0 - \Lambda_t,\ \Lambda_t I\big)$ with $\Lambda_t = \int_0^t \beta(s)ds$. A stable oracle sampler works entirely in log space and *reapplies the original sign* at the end:

1. Store the input sign $s = \mathrm{sign}(x_T)$.
2. Map to log magnitude $y_T = \log(|x_T| + \epsilon)$.
3. Run an additive Gaussian ancestral sampler in log space to obtain y_0.
4. Exponentiate $|x_0| = \exp(y_0)$.
5. Restore sign $x_0 = s \odot |x_0|$.

A numerically correct step for the log–space ancestral sampler uses the Gaussian posterior $Y_{k-1} \mid (Y_k, Y_0)$ with

$$\Delta\Lambda_k = \Lambda_{t_k} - \Lambda_{t_{k-1}}, \quad \begin{cases} \text{prior: } Y_{k-1} \mid Y_0 \sim \mathcal{N}\big(Y_0 - \Lambda_{t_{k-1}},\ \Lambda_{t_{k-1}}\big), \\ \text{likelihood: } Y_k \mid Y_{k-1} \sim \mathcal{N}\big(Y_{k-1} - \Delta\Lambda_k,\ \Delta\Lambda_k$$

Closed–form conditioning gives

$$v_{\text{post}} = \left(\frac{1}{\Lambda_{t_{k-1}}} + \frac{1}{\Delta\Lambda_k}\right)^{-1}, \quad m_{\text{post}} = v_{\text{post}}\left(\frac{Y_0 - \Lambda_{t_{k-1}}}{\Lambda_{t_{k-1}}} + \frac{Y_k + \Delta\Lambda_k}{\Delta\Lambda_k}\right).$$

Sampling $Y_{k-1} = m_{\text{post}} + \sqrt{v_{\text{post}}}z$ with $z \sim \mathcal{N}(0, I)$ yields a stable and unbiased oracle step.

```
import numpy as np

def make_lambda_from_alphas_sqrt(alphas_sqrt):
    # Lambda_k = -2 * log(sqrt(alpha_bar_k))
    return -2.0 * np.log(np.clip(alphas_sqrt, 1e-12, 1.0))

def denoise_multiplicative_oracle(xT, x0, alphas_sqrt, rng=None, eps=1e-6, clip_val=6.0):
    """
    Oracle sampler for multiplicative noise via log-space ancestral steps.
```

```
    Inputs:
      xT           : noisy observation at t_N (shape (...,))
      x0           : clean signal (oracle)
      alphas_sqrt  : array of length N+1 with sqrt(alpha_bar_k)
    Returns:
      x_denoised   : oracle reconstruction at t_0
    """
    if rng is None:
        rng = np.random.default_rng(123)
    xT = np.asarray(xT, dtype=np.float32)
    x0 = np.asarray(x0, dtype=np.float32)

    # 1) Preserve sign and move to log-magnitude
    signs = np.sign(xT + 0.0)
    yk = np.log(np.abs(xT) + eps)
    y0 = np.log(np.abs(x0) + eps)

    # 2) Precompute cumulative Lambda from alphas_sqrt
    Lambda = make_lambda_from_alphas_sqrt(np.asarray(alphas_sqrt, dtype=np.float32))
    N = len(Lambda) - 1

    for k in range(N, 0, -1):
        # Priors and likelihood parameters
        Lam_k   = Lambda[k]
        Lam_km1 = Lambda[k-1]
        dLam    = max(Lam_k - Lam_km1, 1e-12)
        v1      = max(Lam_km1, 1e-12)  # prior variance
        v2      = dLam                 # likelihood variance

        # Posterior of Y_{k-1} | (Y_k, Y_0)
        v_post = 1.0 / (1.0/v1 + 1.0/v2)
        m_post = v_post * ((y0 - Lam_km1)/v1 + (yk + dLam)/v2)

        if k > 1:
            z = rng.standard_normal(size=yk.shape, dtype=np.float32)
            yk = m_post + np.sqrt(v_post, dtype=np.float32) * z
        else:
            yk = m_post

        # Optional numeric guard
        yk = np.clip(yk, -clip_val, clip_val)

    # 3) Map back and restore sign
    mag = np.exp(yk)
    return signs * mag
```

Listing 4. Oracle multiplicative denoising in log space with sign preservation.

Why this is stable. The log transform converts multiplicative noise to additive Gaussian with a known drift $-\beta(t)$. The Kalman–style posterior above uses both (Y_k) and the oracle prior (Y_0), avoiding large drifts and eliminating sign ambiguity by restoring $s = \mathrm{sign}(x_T)$.

4. Impulsive noise: stable oracle correction of the posterior target

For jump noise, the exact reverse intensity is a density–ratio that can be expensive and numerically brittle. An effective *oracle* alternative is to correct the ancestral posterior target $\hat{x}_0$ at the known jump locations, instead of creating an extremely large score near impulses. This yields a stable "oracle inpainting" step inside an otherwise standard additive ancestral sampler.

```
def denoise_jump_oracle(xT, x0, jump_mask, alphas_bar, alphas_sqrt, sigmas, rng=None, clip_val=3.0):
    """
    Oracle sampler for impulsive noise:
      - Use additive ancestral steps.
      - Override the x0-prediction at known jump locations.
    Inputs:
      xT, x0        : noisy and clean arrays (same shape)
      jump_mask     : boolean array (True where impulses occurred)
      alphas_bar    : \bar{alpha}_k array (len N+1)
      alphas_sqrt   : sqrt(\bar{alpha}_k) (len N+1)
      sigmas        : sigma_k (len N+1)
    """
    if rng is None:
        rng = np.random.default_rng(2027)
    xt = np.array(xT, dtype=np.float32, copy=True)
    x0 = np.array(x0, dtype=np.float32, copy=False)
    jm = np.array(jump_mask, dtype=bool, copy=False)

    N = len(alphas_bar) - 1
    for k in range(N, 0, -1):
        # Oracle additive score and noise prediction
        s_oracle = -(xt - alphas_sqrt[k] * x0) / (sigmas[k]**2 + 1e-12)
        eps_pred = -sigmas[k] * s_oracle

        # Posterior x0 estimate (DDPM-style)
        x0_pred = (xt - sigmas[k] * eps_pred) / (alphas_sqrt[k] + 1e-12)

        # Oracle correction only at known jump locations
        x0_pred[jm] = x0[jm]

        # DDPM posterior (mean/variance) for x_{k-1} | (x_k, x0_pred)
        ab_k     = alphas_bar[k]
        ab_km1   = alphas_bar[k-1] if k > 0 else 1.0
        alpha_t  = np.clip(ab_k / (ab_km1 + 1e-12), 1e-6, 0.999999)
```

```
        sqrt_at   = np.sqrt(alpha_t, dtype=np.float32)
        one_m_ab = np.clip(1.0 - ab_k, 1e-12, 1.0)
        one_m_am = np.clip(1.0 - ab_km1, 1e-12, 1.0)
        beta_t    = np.clip(1.0 - alpha_t, 1e-12, 1.0)
        beta_t_tilde = (one_m_am / one_m_ab) * beta_t

        coef_x0  = (np.sqrt(ab_km1) * (1.0 - alpha_t)) / one_m_ab
        coef_xt  = (sqrt_at * one_m_am) / one_m_ab
        mean_post = coef_x0 * np.clip(x0_pred, -clip_val, clip_val) + coef_xt * xt

        if k > 1:
            z = rng.standard_normal(size=xt.shape, dtype=xt.dtype)
            xt = mean_post + np.sqrt(np.clip(beta_t_tilde, 1e-12, 1.0)) * z
        else:
            xt = mean_post
    return xt
```

Listing 5. Stable jump denoising via oracle correction of $\hat{x}_0$ at known impulse locations.

Why this is stable. The update never creates large reverse drifts. The oracle only modifies $\hat{x}_0$ at impulse sites, which are sparse. Everywhere else, the step reduces to the standard additive ancestral update. This isolates impulsive corruption while preserving the Gaussian background.

5. Combined additive and multiplicative noise

A common experimental setting is a *cascade* of heteroscedastic and homoscedastic corruptions. We adopt the forward model

$$x_{\text{mult}} = \Phi_{\text{mult}}(x_0; T_{\text{mult}}), \quad x_{\text{obs}} = x_{\text{mult}} + \epsilon_{\text{add}}, \quad \epsilon_{\text{add}} \approx$$

where Φ_{mult} denotes the geometric VP flow run for horizon T_{mult}. The observation x_{obs} is formed by subsequent additive instrumentation noise. The reverse computation follows the *chain rule of posteriors* and proceeds in the reverse order of the forward corruptions:

1. **Remove additive noise.** From x_{obs}, compute an estimate $\hat{x}_{\text{mult}}$ by an additive ancestral step that targets the intermediate x_{mult}.
2. **Remove multiplicative noise.** From $\hat{x}_{\text{mult}}$, run the log–space oracle sampler to recover $\hat{x}_0$.

In the oracle setting we have access to (x_{mult}, x_0), which yields a numerically stable and verifiable pipeline. In practical (learned) settings, x_{mult} is unknown and the first stage uses a learned score s_θ^{add}; the second stage uses the log–space multiplicative score $\tilde{s}_\theta$.

Schedules. We allow different VP schedules for the additive and multiplicative stages. Let

$$\{\bar{\alpha}_k^{\text{add}},\ \sqrt{\bar{\alpha}_k^{\text{add}}},\ \sigma_k^{\text{add}}\}_{k=0}^{N_{\text{add}}} \quad \text{and} \quad \{\sqrt{\bar{\alpha}_k^{\text{mult}}}\}_{k=0}^{N_{\text{mult}}}$$

be precomputed by `make_vp_schedule` (Listing 1) for the two stages. The multiplicative log–space sampler reconstructs $\Lambda_k^{\text{mult}} = -2\log\sqrt{\bar{\alpha}_k^{\text{mult}}}$ internally.

```
import numpy as np

def denoise_combined_oracle(x_obs,
                            x_mult_oracle,
      # intermediate (oracle)
                            x0_oracle,
      # clean (oracle)
                            add_sched,
      # (alphas_bar, alphas_sqrt, sigmas)
                            mult_sched,
      # (alphas_sqrt) for log-space stage
                            rng=None):
    """
    Two-stage reverse pipeline for a forward cascade: multiplicative -> additive.

    Stage 1 (additive): x_obs -> hat{x_mult} using oracle posterior.
    Stage 2 (multiplicative): hat{x_mult} -> hat{x0} via log-space oracle sampler.

    Inputs:
      x_obs          : observed signal (array)
      x_mult_oracle  : true intermediate (array, oracle target for stage 1)
      x0_oracle      : true clean signal (array, oracle target for stage 2)
      add_sched      : tuple (alphas_bar_add, alphas_sqrt_add, sigmas_add)
      mult_sched     : tuple (alphas_sqrt_mult,)
      rng            : np.random.Generator (optional)
    Returns:
      x0_hat         : oracle reconstruction of x0
    """
    if rng is None:
        rng = np.random.default_rng(7)

    alphas_bar_add, alphas_sqrt_add, sigmas_add = add_sched
    (alphas_sqrt_mult,) = mult_sched

    # ---- Stage 1: remove additive noise (target x_mult_oracle)
    x_mult_hat = denoise_additive_oracle(
        y_noisy      = x_obs,
        x0           = x_mult_oracle,
        alphas_bar   = alphas_bar_add,
```

```
        alphas_sqrt   = alphas_sqrt_add,
        sigmas        = sigmas_add,
        rng           = rng,
        clip_val      = 3.0
    )

    # ---- Stage 2: remove multiplicative
    noise in log space (target x0_oracle)
    x0_hat = denoise_multiplicative_oracle(
        xT            = x_mult_hat,
        x0            = x0_oracle,
        alphas_sqrt   = alphas_sqrt_mult,
        rng           = rng,
        eps           = 1e-6,
        clip_val      = 6.0
    )
    return x0_hat
```

Listing 6. Two-stage oracle pipeline: additive then multiplicative (log space).

Remarks. (i) The order of denoising reverses the forward corruption order. (ii) Stage schedules may use different horizons and step counts; short schedules suffice for stable PF–ODE or ancestral updates. (iii) If signs are present, the second stage's log–space sampler preserves and reapplies them, avoiding sign flips. (iv) In non–oracle pipelines, replace `denoise_additive_oracle` by a learned additive sampler and `denoise_multiplicative_oracle` by its learned log–space counterpart. A light calibration of the additive schedule on real data (e.g., moment matching of residuals) often improves robustness.

This two–stage structure illustrates how score–based components compose under cascaded noise. It also provides a clean harness for ablation: each stage can be validated independently against its oracle target before replacing it with a learned model.

[1] Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. ICLR*, 2021.

[2] U. G. Haussmann and É. Pardoux, "Time reversal of diffusions," *Ann. Probab.* **14**, 1188–1205 (1986).

[3] H. Föllmer, "An entropy approach to the time reversal of diffusion processes," in *Stochastic Differential Systems*, Lecture Notes in Control and Information Sciences **69**, 156–163 (Springer, 1985).

[4] W. J. Anderson, "Reverse-time diffusion equation models," *Z. Wahrscheinlichkeitstheorie verw. Gebiete* **59**, 67–92 (1982).

[5] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.* **23**, 1661–1674 (2011).

[6] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *J. Mach. Learn. Res.* **6**, 695–709 (2005).

[7] L. C. G. Rogers and D. Williams, *Diffusions, Markov Processes and Martingales*, Vol. 2, 2nd ed. (Cambridge Univ. Press, 2000).

[8] I. Karatzas and S. E. Shreve, *Brownian Motion and Stochastic Calculus*, 2nd ed. (Springer, 1991).

[9] K.-I. Sato, *Lévy Processes and Infinitely Divisible Distributions* (Cambridge Univ. Press, 1999).

[10] A. Papapantoleon and C. Xu, "Time reversal of jump diffusions," *Stoch. Proc. Appl.* **143**, 1–40 (2022).

[11] C. Xu, S. Zhou, Y. Tan, and B. Poole, "Efficient score computation for jump diffusion models," in *Advances in Neural Information Processing Systems* 35 (NeurIPS), 2022.

[12] C. Lu, Y. Li, A. Karras, and E. Agustsson, "DPM–Solver: A fast ODE solver for diffusion probabilistic models," arXiv:2206.00927 (2022).

[13] C. Léonard, "A survey of the Schrödinger problem and some of its connections with optimal transport," *Discrete Contin. Dyn. Syst.* **34**, 1533–1574 (2014).

[14] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. (MIT Press, 2018).

[15] Y. Yu and S. T. Acton, "Speckle reducing anisotropic diffusion," *IEEE Trans. Image Process.* **11**, 1260–1270 (2002).

[16] R. L. White, M. Stys, and J. Beckwith, "A robust cosmic-ray rejection algorithm for HST WFC3/UVIS," *Astron. J.* **159**, 46 (2020).

[17] If $p_0 = \mathcal{N}(0, I_d)$ and the schedule is variance preserving, then the marginal p_{t_k} is also $\mathcal{N}(0, I_d)$ and its score equals $-x$. The DSM target (33) is the conditional score; its expectation over x_0 equals the marginal score, but the pointwise targets differ.

[18] The conditional (perturbation) score used for DSM targets differs pointwise from the marginal score, but its population minimizer is the marginal score.