Semimartingale Bracket Descent:
Generator-Normalized Learning Beyond Differentiable Parameter Spaces

Andrew Kiruluta
UC Berkeley CA, School of Information
kiruluta@berkeley.edu

August 28, 2026

Abstract

Standard neural-network training defines learning through derivatives of a scalar objective. We propose *Semimartingale Bracket Descent* (SBD), a derivative-free framework in which trainable states evolve under a known stochastic exploration law and sensitivity is inferred from state–loss quadratic covariation. For a Markov generator $\mathcal{A}$, state embedding ψ, and black-box loss F, the regularized generator-bracket derivative is $D_{\mathcal{A},\varepsilon}F = (\Gamma_{\mathcal{A}}(\psi,\psi)+\varepsilon I)^{-1}\Gamma_{\mathcal{A}}(\psi,F)$. For diffusions it converges, as $\varepsilon \downarrow 0$, to the gradient projected onto the excited subspace when a classical gradient exists; for jump processes it is a nonlocal secant regression; and on finite/countable state spaces it is a weighted graph regression requiring no differentiable parameter trajectory. We state explicit generator-domain and local-integrability assumptions, remove an unnecessary constant-rank condition from the regularized algorithm, and prove that exponential tilting of a discrete jump kernel monotonically decreases the bracket-predicted loss increment, with a sufficient condition for true expected descent under bounded regression error. The practical neural estimator uses paired forward losses, excitation-time exponential averaging, and no backpropagation. On three-seed standard classification benchmarks, SBD reaches $83.33\% \pm 0.83\%$ on Digits and $97.08\% \pm 1.34\%$ on Breast Cancer with zero gradient evaluations, slightly exceeding a matched-query SPSA baseline while remaining less query-efficient than Adam/SGD. A 2,410-parameter MLP and a ReLU ablation expose current scaling limitations. Discrete comparisons against hill climbing, simulated annealing, and the cross-entropy method show that the original binary Hamming task verifies learning relative to random walk but is too easy to establish superiority. The novelty claim is therefore narrow: using normalized generator state–loss energy as a common optimizer primitive across diffusion, Lévy, and discrete Markov learning. Code implementation is at git repo: https://github.com/andrew-jeremy/Stochastic-Gradient-Descent

1 Introduction

Backpropagation converts a scalar objective into a chain of local derivatives and remains the dominant mechanism for training neural networks. Stochastic gradient descent randomizes the derivative estimate; Langevin methods add diffusion around a gradient drift; adaptive optimizers reshape the gradient; and neural SDEs may place stochastic dynamics inside the model. These methods are powerful, but their training signal is still fundamentally a derivative or adjoint sensitivity.

This paper asks a different question:

> *Can deliberately generated stochastic motion of the trainable state itself act as the sensor from which a learning direction is inferred, even when the training trajectory or parameter*

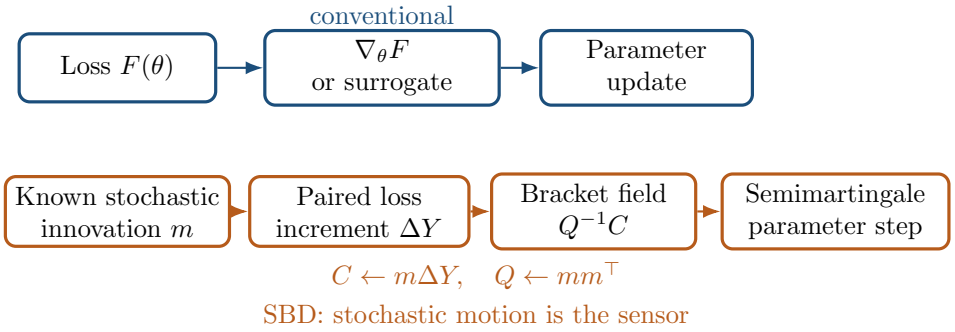

Figure 1: In conventional stochastic optimization, a derivative is computed and noise may then be added. In SBD, stochastic state motion creates the observations from which the bracket field is learned.

space is not differentiable?

We call the resulting framework **Semimartingale Bracket Descent** (SBD). The learner perturbs or transitions the trainable state according to a known Markov exploration law, observes the corresponding scalar loss increment, and estimates a field from state–loss quadratic energy. In the online semimartingale view, if M^Θ and M^Y are the martingale parts of the parameter and loss processes, the local learning signal is determined by

$$\mathrm{d}\left\langle M^\Theta, M^Y \right\rangle_t = \mathrm{d}\left\langle M^\Theta, M^\Theta \right\rangle_t G_t \tag{1}$$

when such a Radon–Nikodym relation exists on the excited subspace. At the generator level, the same idea becomes a normalized carré-du-champ operator and extends naturally to jump processes and finite/countable trainable spaces.

The stochastic motion is not noise added after a gradient has already been computed. It is the measurement process. When F is smooth and the exploration is diffusive, Itô calculus shows that the normalized bracket field converges to the ordinary gradient projected onto the excited directions. When finite jumps are present, exact loss increments across those jumps contribute nonlocal secants. On a discrete Markov state graph, the same normal equations become a weighted regression over transitions, so a differentiable path does not need to exist at all.

1.1 Why stochastic calculus is substantive

With Brownian exploration the parameter path is almost surely nowhere classically differentiable; with a compound-Poisson or more general Lévy component it is càdlàg and may make finite jumps. The appropriate objects are therefore predictable quadratic covariation, jump compensators, generator domains, and local time rather than only ordinary derivatives. Meyer–Tanaka local time remains analytically useful around kinks, but it is a singular finite-variation term and is not a separate quadratic-covariation learning channel.

Figure 1 contrasts SBD with a conventional noisy optimizer.

1.2 What v4 changes

The earlier generator formulation established a conceptual unification but left several practical and mathematical gaps. This revision makes seven substantive changes.

1. **Explicit generator assumptions.** We state the domain and local square-integrability conditions under which $\Gamma_{\mathcal{A}}$ agrees with predictable covariation for vector coordinates and loss.

2. **Regularization without constant rank.** For $\varepsilon > 0$, $Q_{\mathcal{A}}+\varepsilon I$ is positive definite whenever $Q_{\mathcal{A}}$ is finite positive semidefinite; the implemented optimizer therefore does not require a constant-rank hypothesis. Constant-rank issues appear only in the unregularized projection limit.

3. **Excitation-time estimation.** Sparse coordinates update their exponential bracket statistics only when actually excited. Treating an unobserved coordinate as a zero observation is shown empirically to be harmful.

4. **Matched-oracle neural benchmarks.** We compare SBD with SPSA, Adam, and SGD on standard classification datasets, report forward-query and gradient-evaluation budgets, and include nonlinear and ReLU ablations.

5. **Discrete baselines.** Generator-SBD is compared with random walk, greedy hill climbing, simulated annealing, and the cross-entropy method on the same binary task.

6. **Jump-feedback descent analysis.** We prove that exponential tilting monotonically lowers the bracket-predicted transition loss and give a sufficient condition for true expected descent when regression error is controlled.

7. **Exploration-law ablation.** Gaussian, Laplace, and Student-t jump laws are compared; Student-t is reclassified as an optional heavy-tailed design choice rather than a privileged component of SBD.

1.3 Empirical headline and scope

On the three-seed Digits benchmark with a 650-parameter linear classifier, SBD obtains $83.33\% \pm 0.83\%$ test accuracy after 2,000 forward loss evaluations and no gradient evaluations, compared with $80.00\% \pm 2.22\%$ for matched-query SPSA. Adam and SGD reach approximately 95.7–95.9% with 300 gradient evaluations. On the Breast Cancer dataset, SBD obtains $97.08\% \pm 1.34\%$ with 1,000 forward queries, slightly above the matched SPSA result. A 2,410-parameter nonlinear MLP reaches only $72.78\% \pm 1.92\%$, and tensor-cycling blockwise SBD performs worse under the same query budget, exposing rather than resolving the scaling problem.

These results support feasibility, not superiority. The novelty claim is correspondingly narrow: the proposed contribution is the use of a normalized state–loss carré-du-champ/predictable-covariation field as an optimizer primitive that has one definition across diffusion, Lévy, and pure-jump discrete trainable spaces. Stochastic calculus, carré-du-champ theory, SPSA, derivative-free optimization, simulated annealing, and the cross-entropy method are established prior art.

2 Related Work and Novelty Boundary

SBD sits at the intersection of stochastic approximation, derivative-free optimization, stochastic-process calculus, and discrete stochastic search. The contribution is therefore a combination claim, and the distinctions from nearby methods must be explicit.

2.1 Stochastic gradients and Langevin methods

SGLD augments stochastic-gradient updates with Gaussian noise [1], while nonconvex Langevin analyses and recent zeroth-order variants replace the drift with stochastic or finite-difference gradient estimates [2–4]. These methods make the optimization trajectory stochastic, but a gradient-like drift is supplied or explicitly approximated before the SDE is formed. SBD instead defines its field from the quadratic energy generated by the stochastic exploration process itself.

2.2 SPSA, evolution strategies, and derivative-free optimization

Spall's simultaneous perturbation stochastic approximation (SPSA) obtains a stochastic gradient approximation from two noisy function evaluations independent of dimension [5]. Evolution strategies, direct search, Gaussian smoothing, importance-sampled derivative-free schemes, and random-subspace methods likewise optimize black-box or non-differentiable objectives [8–13]. SBD therefore does not claim novelty from using random perturbations or two forward losses.

The finite-sample SBD regression $Q^{-1}C$ can resemble a zeroth-order estimator on smooth Euclidean objectives. The claimed distinction is structural: Q and C estimate the predictable energy of a specified Markov generator, and the same object remains defined for finite jumps and graph transitions where neither an infinitesimal Euclidean perturbation nor a smoothed-objective gradient is required. The empirical section includes SPSA specifically because it is the closest classical two-query baseline.

2.3 Discrete stochastic optimization

The pure-jump formulation must also be compared with mature discrete search. Simulated annealing accepts uphill proposals according to a temperature-controlled Metropolis rule and has a long history in combinatorial optimization [6]. The cross-entropy method adapts a proposal distribution using elite samples and applies to combinatorial, continuous, and noisy optimization [7]. Greedy hill climbing and random walk are simpler local baselines. Generator-SBD differs by estimating a state–loss secant regression from realized Markov transitions and using that regression to tilt future transition probabilities. This organization is novel only if it yields useful learning behavior beyond what established discrete search already provides; accordingly, Section 9 reports all four baselines and shows that the original separable Hamming task is too easy to demonstrate an advantage over hill climbing, annealing, or CEM.

2.4 Consensus and stochastic-control methods

Consensus-based optimization uses interacting derivative-free particles and loss-weighted consensus [14]. Path-integral control uses trajectory weighting to improve controls under stochastic-control structure [15]. These methods establish that stochastic trajectories themselves can be algorithmic objects. SBD differs by fitting a normalized local/nonlocal state–loss regression in the generator energy geometry and feeding the resulting field back into either continuous drift or jump intensities.

2.5 Neural SDEs and jump-diffusion models

Neural SDE and jump-diffusion models place stochastic dynamics inside the forward model, while trainable parameters are generally updated using backpropagation, adjoints, or gradient-based control formulations; recent jump-diffusion neural-network work remains in this family [16]. SBD instead makes the trainable state itself the explored stochastic process and does not require differentiation through the black-box objective.

Table 1: Conceptual positioning. "Forward-only" means that the optimizer can be run without reverse-mode differentiation through the objective.

Method	Forward-only	Stochastic state	Discrete native	Energy-normalized	Primary signal
SGD/Adam	no	optional	no	no	backprop gradient
SGLD	no	yes	no	no	gradient + diffusion
SPSA	yes	probes	no	no	two-sided perturbation gradient
ES	yes	population	indirect	no	fitness-weighted perturbations
Simulated annealing	yes	yes	yes	no	Metropolis energy difference
Cross-entropy method	yes	population	yes	no	elite-distribution update
Generator-SBD	yes	yes	yes	yes	normalized state–loss bracket

2.6 Martingale covariation and carré-du-champ methods

Guo, Tang, and Xu use martingale and covariation identities to estimate a conditioning function and its gradient for hard-constrained diffusion guidance [17]. This is a particularly close conceptual precedent: it demonstrates that quadratic-variation identities can reveal gradient information from stochastic trajectories. Their goal, however, is guidance of a pretrained diffusion sampler rather than a general parameter optimizer spanning continuous and discrete trainable spaces.

Carré-du-champ operators and energy measures are classical in Markov-process and Dirichlet-form theory [19, 20]. Recent carré-du-champ flow matching uses the operator to construct geometry-aware noise in generative modeling [21]. SBD does not claim invention of carré-du-champ calculus. The proposed optimizer-level contribution is the normalized cross-energy

$$(\Gamma_{\mathcal{A}}(\psi,\psi)+\varepsilon I)^{-1}\Gamma_{\mathcal{A}}(\psi,F) \tag{2}$$

as the learning field, together with feedback use across diffusion, jump-diffusion, and pure-jump trainable state spaces.

2.7 Novelty matrix

The defensible novelty statement is therefore deliberately narrow. SBD proposes a common optimizer definition based on normalized generator state–loss energy and a feedback interpretation of that field. It does not claim that stochasticity, forward-only optimization, covariation identities, carré-du-champ theory, or discrete stochastic search are new.

3 Learning as a Parameter Semimartingale

Let a model f_θ be parameterized by $\theta \in \mathbb{R}^d$. For a minibatch or data sample ξ, define a scalar black-box loss

$$F(\theta;\xi) \in \mathbb{R}, \tag{3}$$

and population risk $\bar{F}(\theta) = \mathbb{E}_\xi[F(\theta;\xi)]$ when the expectation exists. SBD does not assume that F is differentiable with respect to θ. The forward computation itself may contain hard thresholds, quantization, discrete routing, external simulators, combinatorial modules, or non-differentiable system calls.

3.1 Continuous-time idealization

We model the trainable parameter state as a càdlàg semimartingale

$$\boxed{\mathrm{d}\boldsymbol{\Theta}_t = -A_t \boldsymbol{G}_t \,\mathrm{d}t + \boldsymbol{\Sigma}_t \,\mathrm{d}W_t + \int_{\mathcal{Z}} \boldsymbol{\Gamma}_t(z)\, \widetilde{N}(\mathrm{d}t, \mathrm{d}z),} \tag{4}$$

where:

- $A_t \succeq 0$ is a learning preconditioner or scalar gain;
- W_t is a Brownian motion generating continuous exploration;
- $\boldsymbol{\Sigma}_t$ specifies the explored subspace and anisotropy;
- $\widetilde{N}$ is a compensated Poisson random measure with base Lévy measure $\nu(\mathrm{d}z)$;
- $\boldsymbol{\Gamma}_t(z)$ maps jump marks to finite parameter displacements;
- $\boldsymbol{G}_t$ is the bracket field estimated from observed co-movement of parameters and loss.

The loss process is

$$Y_t = F(\boldsymbol{\Theta}_t; \xi_t), \tag{5}$$

with the important implementation convention that paired loss differences use the same minibatch or common random numbers. This suppresses nuisance variance from changing data while preserving stochastic parameter motion.

Equation (4) is not meant to imply that training literally needs an infinitesimal-time solver. It is an organizing limit from which a practical Euler scheme is derived. Its purpose is to identify the correct stochastic differential object.

3.2 Continuous and jump brackets

For scalar semimartingales X and Y, quadratic covariation decomposes as

$$[X, Y]_t = [X^c, Y^c]_t + \sum_{0<s\le t} \Delta X_s \Delta Y_s. \tag{6}$$

For vector $\boldsymbol{\Theta}$, the matrix quadratic variation and vector cross-variation are

$$[\boldsymbol{\Theta}]_t := \big([\Theta^i, \Theta^j]_t\big)_{i,j=1}^{d}, \tag{7}$$

$$[\boldsymbol{\Theta}, Y]_t := \big([\Theta^i, Y]_t\big)_{i=1}^{d}. \tag{8}$$

Thus one stochastic object automatically records both local Brownian information and finite jump information.

3.3 The bracket field as a stochastic derivative

Suppose the predictable covariation measures are absolutely continuous with respect to a common increasing clock. Informally, the SBD field is the matrix-measure quotient

$$\boxed{\boldsymbol{G}_t^{(\varepsilon)} = \left(\frac{\mathrm{d}\left\langle M^\Theta, M^\Theta\right\rangle_t}{\mathrm{d}t} + \varepsilon I\right)^{-1} \frac{\mathrm{d}\left\langle M^\Theta, M^Y\right\rangle_t}{\mathrm{d}t},} \tag{9}$$

where M^Θ and M^Y denote martingale components and $\varepsilon > 0$ stabilizes poorly excited directions. The Moore–Penrose inverse is used only for the unregularized $\varepsilon \downarrow 0$ projection limit.

This resembles a stochastic regression coefficient: the numerator records how loss co-moves with parameter innovations, and the denominator normalizes by how much stochastic motion occurred in each direction. Crucially, no classical derivative appears in the definition.

3.4 Geometric interpretation

The diffusion covariance

$$a_t = \mathbf{\Sigma}_t \mathbf{\Sigma}_t^\top \tag{10}$$

acts as an exploration metric. When a_t is rank deficient, SBD only identifies sensitivity in the range of a_t. This is a feature rather than a defect for scalable training: one may excite a layer, block, random subspace, low-rank adapter, or structured transform at a time and rotate those subspaces across training.

Here "energy means quadratic-variation/carr'e-du-champ energy induced by the exploration process; it is not a physical energy or a separately chosen metric. The learning geometry therefore has two coupled components:

1. an *exploration geometry* determined by quadratic variation; and
2. a *descent geometry* obtained by regressing loss variation against that stochastic geometry.

Unlike an externally chosen natural-gradient metric, the bracket metric is generated by the actual stochastic path used to probe the model.

4 The Bracket Derivative and the Smooth Limit

This section establishes the central consistency result: if an ordinary gradient exists, SBD recovers it from quadratic covariation.

Definition 4.1 (Continuous bracket derivative)**.** *Let $\mathbf{\Theta}_t \in \mathbb{R}^d$ be a continuous semimartingale with martingale part M^Θ, and let Y_t be a scalar semimartingale with martingale part M^Y. If there is a predictable vector process G_t such that*

$$\mathrm{d}\left\langle M^\Theta, M^Y \right\rangle_t = \mathrm{d}\left\langle M^\Theta, M^\Theta \right\rangle_t G_t, \tag{11}$$

then G_t is called a bracket derivative *of Y with respect to $\mathbf{\Theta}$ on the excited subspace.*

The definition is analogous to a Radon–Nikodym derivative between covariation measures. It is also closely related to the coefficient appearing in martingale representation and Kunita–Watanabe decompositions, but here it is interpreted operationally as a learning direction.

Theorem 4.2 (Smooth recovery)**.** *Let $F \in C^1(\mathbb{R}^d)$ and suppose*

$$\mathrm{d}\mathbf{\Theta}_t = b_t \,\mathrm{d}t + \mathbf{\Sigma}_t \,\mathrm{d}W_t \tag{12}$$

with $a_t = \mathbf{\Sigma}_t \mathbf{\Sigma}_t^\top$. Let $Y_t = F(\mathbf{\Theta}_t)$ and assume the required local integrability. Then

$$\frac{\mathrm{d}\left\langle M^\Theta, M^Y \right\rangle_t}{\mathrm{d}t} = a_t \nabla F(\mathbf{\Theta}_t). \tag{13}$$

Consequently,

$$G_t = a_t^\dagger a_t \nabla F(\mathbf{\Theta}_t). \tag{14}$$

If a_t is positive definite, $G_t = \nabla F(\mathbf{\Theta}_t)$.

Proof. By Itô's formula, the martingale component of Y is

$$\mathrm{d}M_t^Y = \nabla F(\boldsymbol{\Theta}_t)^\top \boldsymbol{\Sigma}_t \, \mathrm{d}W_t. \tag{15}$$

The martingale component of $\boldsymbol{\Theta}$ is $\mathrm{d}M_t^\Theta = \boldsymbol{\Sigma}_t \mathrm{d}W_t$. Therefore

$$\mathrm{d}\left\langle M^\Theta, M^Y \right\rangle_t = \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_t^\top \nabla F(\boldsymbol{\Theta}_t)\, \mathrm{d}t = a_t \nabla F(\boldsymbol{\Theta}_t)\, \mathrm{d}t, \tag{16}$$

while $\mathrm{d}\left\langle M^\Theta, M^\Theta \right\rangle_t = a_t \mathrm{d}t$. Applying the pseudoinverse gives the result. □

The theorem exposes the conceptual reversal. In a differentiable problem, the gradient can be *derived from* the stochastic bracket. In SBD, the bracket is observable from path increments even when the symbolic gradient is unavailable or the forward graph is non-differentiable.

4.1 Discrete bracket regression

Let m_k be a known stochastic innovation applied to parameters over a short time interval and let

$$\Delta Y_k = F(\theta_k + m_k; \xi_k) - F(\theta_k; \xi_k) \tag{17}$$

use the same minibatch ξ_k. Over a window $\mathcal{W}$ define

$$\widehat{Q} = \sum_{k \in \mathcal{W}} m_k m_k^\top, \tag{18}$$

$$\widehat{C} = \sum_{k \in \mathcal{W}} m_k \Delta Y_k, \tag{19}$$

$$\widehat{G} = (\widehat{Q} + \varepsilon I)^{-1} \widehat{C}. \tag{20}$$

This is the finite-sample bracket estimator. No division by an individual perturbation size is required; the normalization is performed by the accumulated quadratic variation.

Proposition 4.3 (Small-noise local consistency)**.** *Assume $F \in C^2$ near θ, m is mean-zero with covariance $Q = \mathbb{E}[mm^\top]$, and $\mathbb{E}\|m\|^3 < \infty$. Then*

$$\mathbb{E}[m\{F(\theta + m) - F(\theta)\}] = Q \nabla F(\theta) + O(\mathbb{E}\|m\|^3). \tag{21}$$

If Q is nonsingular, the bracket regression recovers $\nabla F(\theta)$ up to the corresponding higher-order perturbation error.

Proof. Taylor expansion gives

$$F(\theta + m) - F(\theta) = \nabla F(\theta)^\top m + \tfrac{1}{2} m^\top H(\theta) m + O(\|m\|^3). \tag{22}$$

Multiplying by m and taking expectations yields the leading term $Q \nabla F(\theta)$. For symmetric innovations the cubic contribution from the Hessian term vanishes, improving the leading bias. □

4.2 Diagonal estimator

Storing a dense $d \times d$ bracket matrix is impossible for modern networks. Under dense excitation the simplest approximation keeps only diagonal quadratic variation,

$$C_{k,i} = \rho C_{k-1,i} + (1-\rho) m_{k,i} \Delta Y_k, \tag{23}$$

$$Q_{k,i} = \rho Q_{k-1,i} + (1-\rho) m_{k,i}^2, \tag{24}$$

$$\widehat{G}_{k,i} = \frac{C_{k,i}}{Q_{k,i} + \varepsilon}. \tag{25}$$

The state cost is $2d$ scalars, comparable to Adam's first- and second-moment buffers, but the buffers summarize state–loss and state–state stochastic variation rather than gradients. For *sparse* exploration, Equation (25) must not decay an unexcited coordinate as though a zero bracket observation had been received. Section 7 therefore uses an excitation-time EMA whose local clock advances only when $m_{k,i} \neq 0$.

A richer low-rank variant can maintain Q only inside a rotating r-dimensional block or random subspace. This creates a tunable continuum between one-direction stochastic probing and full local identification.

5 Generator-Normalized Bracket Calculus

The pathwise bracket formulation becomes more general when written at the level of a Markov generator. This section states the domain assumptions explicitly and defines vector/matrix quantities entrywise.

5.1 Generator domain and predictable energy

Let $X = (X_t)_{t \geq 0}$ be a càdlàg Markov semimartingale on a measurable state space E with extended generator $\mathcal{A}$. Let $\psi = (\psi_1, \ldots, \psi_p) : E \to \mathbb{R}^p$ be an embedding of the trainable state and let $F : E \to \mathbb{R}$ be the black-box loss. We assume locally that

$$\psi_i,\ F,\ \psi_i \psi_j,\ \psi_i F \in \mathcal{D}_{\mathrm{loc}}(\mathcal{A}), \tag{26}$$

for all i, j, and that the associated Dynkin local martingales are locally square integrable after stopping. These conditions can be weakened in specific process classes, but they make the covariation identities below unambiguous.

For scalar f, g satisfying the domain condition, define

$$\Gamma_{\mathcal{A}}(f,g) := \mathcal{A}(fg) - f\,\mathcal{A}g - g\,\mathcal{A}f. \tag{27}$$

With this convention, the Dynkin martingales

$$M_t^f = f(X_t) - f(X_0) - \int_0^t \mathcal{A}f(X_s)\,\mathrm{d}s \tag{28}$$

obey, after localization,

$$\left\langle M^f, M^g \right\rangle_t = \int_0^t \Gamma_{\mathcal{A}}(f,g)(X_s)\,\mathrm{d}s. \tag{29}$$

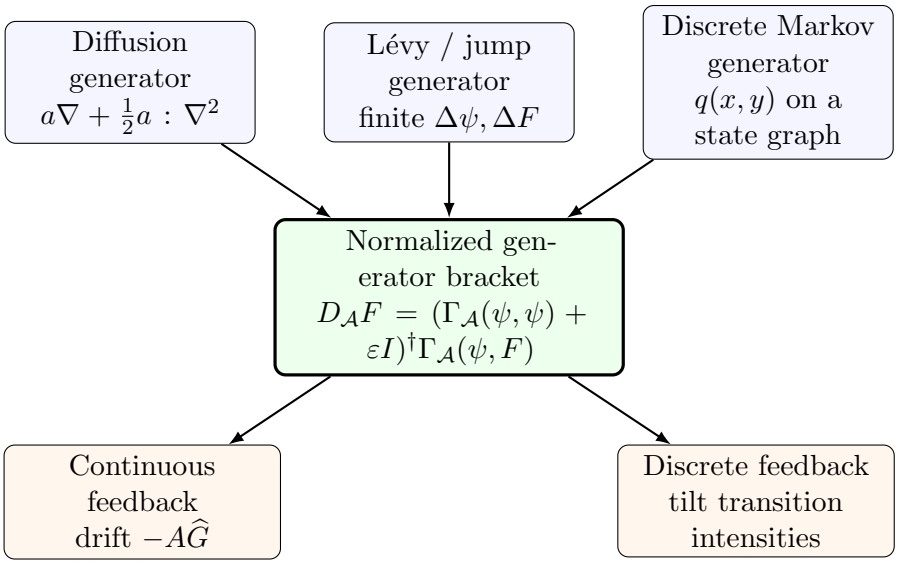

Figure 2: Generator-level unification. Diffusion, finite Lévy jumps, and pure-jump graph transitions contribute to the same normalized state–loss energy field.

The vector and matrix objects used by SBD are defined entrywise:

$$[Q_{\mathcal{A}}(x)]_{ij} := \Gamma_{\mathcal{A}}(\psi_i, \psi_j)(x), \tag{30}$$

$$[C^F_{\mathcal{A}}(x)]_i := \Gamma_{\mathcal{A}}(\psi_i, F)(x). \tag{31}$$

Thus no scalar generator is being applied vaguely to a vector or matrix; every component is an ordinary scalar carré-du-champ.

Definition 5.1 (Regularized generator-bracket derivative)**.** *For $\varepsilon > 0$ and finite positive-semidefinite $Q_{\mathcal{A}}(x)$, define*

$$\boxed{D_{\mathcal{A},\varepsilon}F(x) := \big(Q_{\mathcal{A}}(x) + \varepsilon I\big)^{-1} C^F_{\mathcal{A}}(x).} \tag{32}$$

Because $Q_{\mathcal{A}} \succeq 0$, the regularized matrix is positive definite for every $\varepsilon > 0$. Consequently the implemented optimizer does *not* require a constant-rank assumption. The Moore–Penrose inverse enters only in the unregularized limit $\varepsilon \downarrow 0$ when one wishes to interpret the result as a projection onto the excited range.

5.2 Jump-diffusion decomposition

Consider a Euclidean jump diffusion with diffusion covariance $a(x) = \Sigma(x)\Sigma(x)^{\top}$ and jump displacement $\gamma(x, z)$ under Lévy kernel $\nu_x(\mathrm{d}z)$. Taking $\psi(x) = x$ and assuming the required integrability,

$$C^F_{\mathcal{A}}(x) = a(x)\nabla F(x) + \int \gamma(x, z)\big[F(x + \gamma(x, z)) - F(x)\big]\nu_x(\mathrm{d}z), \tag{33}$$

$$Q_{\mathcal{A}}(x) = a(x) + \int \gamma(x, z)\gamma(x, z)^{\top}\nu_x(\mathrm{d}z). \tag{34}$$

For unbounded jump laws, the displayed integrals are interpreted only when finite or after truncation/localization; SBD does not assume that arbitrary heavy-tailed objectives automatically satisfy these conditions.

Theorem 5.2 (Regularized diffusion and jump limits)**.** *Suppose Equations (33)–(34) are finite.*

1. *If the jump kernel is zero and $F \in C^1$, then*

$$D_{\mathcal{A},\varepsilon}F = (a + \varepsilon I)^{-1} a \nabla F. \tag{35}$$

As $\varepsilon \downarrow 0$, this converges pointwise to $a^\dagger a \nabla F$, the gradient projected onto the excited range. No constant-rank assumption is needed for the fixed-ε identity.

2. *If the diffusion covariance is zero, then*

$$D_{\mathcal{A},\varepsilon}F = \left(\int \gamma\gamma^\top \nu(\mathrm{d}z) + \varepsilon I \right)^{-1} \int \gamma \, \Delta_\gamma F \, \nu(\mathrm{d}z), \tag{36}$$

which is a finite-scale secant field and requires no classical derivative of F.

Proof. Substitute the absent component into Equations (33)–(34). For the diffusion limit, diagonalize $a = U\Lambda U^\top$; $(a + \varepsilon I)^{-1} a$ has eigenvalues $\lambda_i/(\lambda_i + \varepsilon)$ and therefore converges to the orthogonal projector onto range(a). □

5.3 Pure-jump discrete spaces

Let E be finite or countable with jump intensities $q(x, y)$, and let $\psi : E \to \mathbb{R}^p$ encode a discrete model state such as a binary mask, quantized block, router assignment, architecture choice, or symbolic component. For absolutely summable terms,

$$Q_{\mathcal{A}}(x) = \sum_{y \neq x} q(x, y) \, \delta\psi_{xy} \delta\psi_{xy}^\top, \tag{37}$$

$$C_{\mathcal{A}}^F(x) = \sum_{y \neq x} q(x, y) \, \delta\psi_{xy} \, \delta F_{xy}, \tag{38}$$

where $\delta\psi_{xy} = \psi(y) - \psi(x)$ and $\delta F_{xy} = F(y) - F(x)$.

Proposition 5.3 (Graph-space ridge regression)**.** *For $\varepsilon > 0$, $D_{\mathcal{A},\varepsilon}F(x)$ is the unique minimizer*

$$\arg\min_g \left\{ \sum_{y \neq x} q(x, y) \big(\delta F_{xy} - g^\top \delta\psi_{xy} \big)^2 + \varepsilon \|g\|^2 \right\}. \tag{39}$$

Proof. The normal equations are $(Q_{\mathcal{A}} + \varepsilon I)g = C_{\mathcal{A}}^F$, whose coefficient matrix is positive definite. □

The discrete result is important conceptually: the learning field is defined through transitions and loss differences even when no differentiable parameter curve or tangent bundle is available.

5.4 Feedback control of jump probabilities

For a finite set of candidate transitions from state x, let $p_0(y \mid x) > 0$ be normalized base probabilities and define the bracket-predicted increment

$$s_y = G(x)^\top \delta\psi_{xy}. \tag{40}$$

SBD uses the exponential tilt

$$p_\beta(y \mid x) = \frac{p_0(y \mid x) e^{-\beta s_y}}{\sum_z p_0(z \mid x) e^{-\beta s_z}}, \qquad \beta \geq 0, \tag{41}$$

optionally mixed with a small uniform component to preserve exploration. The total jump clock may be set separately; Equation (41) controls the destination distribution. Section 8 gives an exact monotonicity result and a sufficient condition for expected true-loss descent.

6 Beyond Differentiable Learning Trajectories

The main motivation for SBD is not to reproduce gradients inefficiently where backpropagation already works. Its more distinctive regime is learning when classical derivatives are absent, misleading, or operationally inaccessible.

6.1 Nonsmooth losses and local time

For one-dimensional convex functions, generalized Itô calculus provides a precise statement. Let X_t be a continuous semimartingale and F a difference of two convex functions. The Meyer–Tanaka formula can be written schematically as

$$F(X_t) = F(X_0) + \int_0^t F'_-(X_s)\,\mathrm{d}X_s + \frac{1}{2}\int_{\mathbb{R}} L_t^a(X)\,F''(\mathrm{d}a), \tag{42}$$

where F'_- is the left derivative, F'' is the distributional second-derivative measure, and $L_t^a(X)$ is local time.

The local-time term has finite variation and therefore does not contribute to continuous quadratic covariation with X. This yields the following result.

Theorem 6.1 (Nonsmooth one-dimensional bracket recovery)**.** *Under the conditions above,*

$$\mathrm{d}[X, F(X)]_t^c = F'_-(X_t)\,\mathrm{d}[X]_t^c \tag{43}$$

for the continuous quadratic-covariation measures. Therefore, wherever $\mathrm{d}[X]^c$ *provides excitation, the bracket derivative recovers a valid one-sided first-order representative even though* F *need not be classically differentiable at all points.*

The theorem is important conceptually: nondifferentiability does not make stochastic sensitivity meaningless. Singular curvature is represented by local time rather than by an ordinary Hessian. In higher dimensions the geometry is more involved; one can work along one-dimensional slices, Sobolev derivatives, or generalized Itô formulas under additional regularity assumptions. We treat a general multidimensional nonsmooth theory as an open problem rather than overstate it.

6.2 Why Brownian paths help rather than hurt

A Brownian parameter path is almost surely nowhere differentiable. Classical trajectory velocity is therefore the wrong learning object. Quadratic variation is specifically designed to retain second-order information from such paths. SBD exploits this: the noise creates a non-vanishing stochastic differential scale, and parameter–loss co-variation survives in the limit even though ordinary pointwise velocities do not.

6.3 Jump calculus for plateaus and discontinuities

Brownian motion is local. On a large flat plateau, small continuous perturbations can produce $\Delta Y = 0$ for long periods, causing the bracket field to vanish. To create a genuinely nonlocal channel, SBD allows a finite-activity jump component. At a jump time t,

$$\Delta\boldsymbol{\Theta}_t = \boldsymbol{\Gamma}_t(z), \tag{44}$$

$$\Delta Y_t = F(\boldsymbol{\Theta}_{t-} + \boldsymbol{\Gamma}_t(z)) - F(\boldsymbol{\Theta}_{t-}). \tag{45}$$

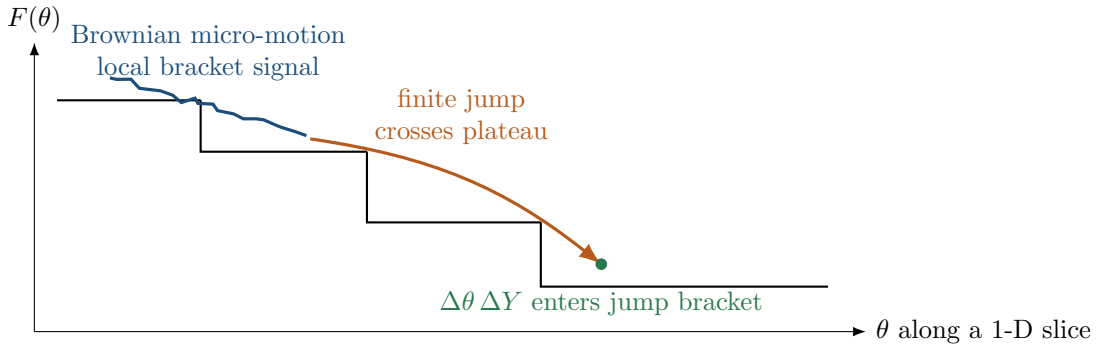

Figure 3: SBD parameter motion can combine local Brownian exploration with finite Lévy-style jumps. The latter can cross flat or discontinuous regions that provide no local signal.

The jump contribution to the cross-bracket is exactly

$$\Delta[\boldsymbol{\Theta}, Y]_t = \Delta\boldsymbol{\Theta}_t\,\Delta Y_t. \tag{46}$$

No limiting derivative appears. A finite displacement can cross a hard threshold, a quantization boundary, a discrete architecture switch, or a flat basin and still contribute a learning signal.

Figure 3 illustrates the combined geometry.

6.4 Jump bracket as nonlocal secant regression

Over a window containing jump innovations j_k, the same regression form

$$\widehat{G}_J = \Big(\sum j_k j_k^\top + \varepsilon I\Big)^{-1} \sum j_k\{F(\theta_k + j_k) - F(\theta_k)\} \tag{47}$$

acts as a scale-dependent, nonlocal secant field. For small jumps and a smooth objective it approaches the local gradient. For large jumps it measures a different object: the best linear predictor of finite loss changes from finite parameter displacements under the chosen jump law.

This distinction is intentional. On discontinuous objectives there may be no useful infinitesimal derivative to recover. A finite-scale stochastic derivative is then more relevant than a vanishing local one.

6.5 Choice of Lévy law

The reference implementation uses a compound-Poisson clock with Student-t jump amplitudes. This is computationally simple and gives heavier tails than Gaussian exploration. Other possibilities include Laplace jumps, sparse coordinate jumps, symmetric α-stable laws, layerwise reset proposals, low-rank adapter jumps, or learned jump kernels. The law should be treated as an exploration design choice, analogous to choosing a proposal distribution in Monte Carlo methods.

6.6 Optional local-time diagnostics

Although the core optimizer does not require explicit numerical local-time estimation, local time can be used diagnostically. An occupation-density estimate of the loss process near a level a is

$$\widehat{L}_T^a(Y;\epsilon) \propto \frac{1}{2\epsilon}\sum_k \mathbf{1}\{|Y_k - a| < \epsilon\}\,\Delta[Y]_k. \tag{48}$$

Large occupation near a best-so-far level may justify annealing diffusion; repeated occupation near abrupt loss transitions may indicate a nonsmooth boundary. A future adaptive SBD variant could use such statistics to control diffusion scale and jump intensity.

7 Practical Semimartingale Bracket Descent

We now turn the stochastic-calculus construction into executable forward-only optimizers.

7.1 Retained stochastic update

At iteration k, with parameters θ_k and a fixed minibatch or simulator seed ξ_k:

1. Evaluate $y_k = F(\theta_k; \xi_k)$.
2. Draw a stochastic innovation
$$m_k = \sigma_k \sqrt{h}\, P_k z_k + J_k, \tag{49}$$
where P_k selects an excited block/subspace and J_k is an optional finite jump.
3. *Retain* that transition and evaluate
$$\widetilde{y}_k = F(\theta_k + m_k; \xi_k), \qquad \Delta y_k = \widetilde{y}_k - y_k. \tag{50}$$
4. Update bracket statistics from the realized products $m_k \Delta y_k$ and $m_k m_k^\top$.
5. Apply the learned drift correction
$$\theta_{k+1} = \theta_k + m_k - \eta_k h A_k \widehat{G}_k. \tag{51}$$

The reference neural implementation uses two forward loss evaluations per SBD step and invokes neither `backward()` nor `autograd.grad()`.

7.2 Excitation-time diagonal estimator

The previous sparse implementation used an ordinary global-step EMA,

$$C_{k,i} = \rho C_{k-1,i} + (1-\rho) m_{k,i} \Delta y_k, \tag{52}$$

even when coordinate i was not excited and therefore $m_{k,i} = 0$. Under sparse exploration this incorrectly treats a missing observation as evidence of zero cross-energy and can rapidly erase useful statistics.

The v4 implementation instead advances the stochastic clock of coordinate i only on its excitation times. Let $I_{k,i} = \mathbf{1}\{m_{k,i} \neq 0\}$. Then

$$C_{k,i} = \begin{cases} \rho C_{k-1,i} + (1-\rho) m_{k,i} \Delta y_k, & I_{k,i} = 1, \\ C_{k-1,i}, & I_{k,i} = 0, \end{cases} \tag{53}$$

$$Q_{k,i} = \begin{cases} \rho Q_{k-1,i} + (1-\rho) m_{k,i}^2, & I_{k,i} = 1, \\ Q_{k-1,i}, & I_{k,i} = 0, \end{cases} \tag{54}$$

$$\widehat{G}_{k,i} = \frac{C_{k,i}}{Q_{k,i} + \varepsilon}. \tag{55}$$

This is an EMA indexed by the coordinate's own observation count rather than the global optimizer step. In the checked Digits configuration, this correction materially improves accuracy at the same forward-query budget.

7.3 Sparse martingale innovations

For high-dimensional neural parameters, the benchmark uses sparse Rademacher innovations

$$m_{k,i} = \sigma_k \sqrt{h}\, \frac{M_{k,i} Z_{k,i}}{\sqrt{p}}, \qquad M_{k,i} \sim \text{Bernoulli}(p), \quad Z_{k,i} \in \{-1, +1\}. \tag{56}$$

Then $\mathbb{E}[m_{k,i}^2] = \sigma_k^2 h$ independently of p, while only a fraction p of coordinates is perturbed per step. This does not remove the zeroth-order curse of dimensionality, but it reduces simultaneous-coordinate interference and works naturally with the excitation-time bracket clock.

7.4 Common random numbers do not imply covariate-shift bias

Paired evaluations should use the same minibatch or simulator randomness. Suppose ξ is independent of the innovation m and define the population loss $\bar{F}(\theta) = \mathbb{E}_\xi F(\theta; \xi)$. Then

$$\mathbb{E}_{m,\xi}\left[m\{F(\theta + m; \xi) - F(\theta; \xi)\}\right] \tag{57}$$

$$= \mathbb{E}_m\left[m\{\bar{F}(\theta + m) - \bar{F}(\theta)\}\right]. \tag{58}$$

Thus same-minibatch pairing does not by itself bias the target cross-moment when sampling is exogenous; it removes nuisance variation that would otherwise enter the loss difference. Bias can arise if the data/simulator sampling law depends on the proposal, current parameter, or other endogenous state. In that case importance weighting or an explicitly state-dependent risk formulation is required.

7.5 Blockwise and low-rank variants

The benchmark includes three practical choices:

Full sparse SBD.
Every tensor is eligible but only a sparse set of elements is excited.

Tensor-cycling SBD.
One parameter tensor is excited at a time, cycling across tensors. This reduces instantaneous dimension but also reduces how often each tensor receives information.

Random-subspace SBD.
In principle one may use $m = P_k u$ with $P_k \in \mathbb{R}^{d \times r}$, estimate an $r \times r$ bracket, and map the field back through P_k. The current neural benchmark implements tensor blocks rather than a dense random-subspace matrix.

The empirical scaling ablation shows that tensor cycling is *not* automatically superior under a fixed query budget; it trades instantaneous dimension against observation frequency.

7.6 Jump-law choice

The reference implementation now supports Gaussian, variance-normalized Laplace, and Student-t jump amplitudes. Student-t was originally chosen to provide occasional long moves across plateaus, but it is not mathematically privileged by SBD. Section 9 reports an ablation showing that Gaussian jumps are better on one plateau task, whereas Laplace/Student-t are better on the hard-threshold task. The exploration law must therefore be treated as a proposal-design hyperparameter.

7.7 Hyperparameter guidelines

The principal black-box hyperparameters have distinct roles:

ε. Ridge regularization of the bracket inverse. A robust default is to set it to a small fraction of the median active diagonal Q_i once a warm-up window is available, e.g. $\varepsilon = 10^{-2}\,\mathrm{median}(Q_i)$ with a numerical floor. Increasing ε shrinks the field in poorly excited directions.

σ. Exploration scale. Initialize so that paired losses change measurably but are not dominated by clipping or catastrophic jumps. A practical black-box calibration is to sample 10–50 pilot perturbations and choose σ so the median $|\Delta F|$ is roughly 1–10% of the current loss scale, then anneal slowly.

β. Discrete generator-tilt inverse temperature. Start near zero and increase only when the transition regression has predictive signal. The theorem in Section 8 guarantees monotonic improvement only for the *predicted* increment; overly large β is unsafe under high regression error.

ρ. Excitation-time memory. $\rho = 0.9$ corresponds to an effective window of roughly ten *excitations per coordinate*; $\rho = 0.99$ corresponds to roughly one hundred. This is distinct from smoothing over global training steps.

p. Sparse element excitation probability. Smaller p reduces simultaneous interference but increases the number of steps needed for all coordinates to receive observations. It should therefore be reported jointly with the query budget.

7.8 Pseudocode

Listing 1: Excitation-time diagonal SBD used in the neural benchmarks.

```
initialize theta, C = 0, Q = eps
for minibatch B:
    y0 = loss(theta, B)                    # forward only
    mask ~ Bernoulli(p)
    z    ~ Rademacher
    m = sigma*sqrt(dt)*mask*z/sqrt(p)
    theta += m                             # retain stochastic move
    y1 = loss(theta, B)                    # same B
    dY = y1 - y0

    active = (m != 0)
    C[active] = rho*C[active] + (1-rho)*m[active]*dY
    Q[active] = rho*Q[active] + (1-rho)*m[active]**2
    G = C / (Q + eps)
    theta -= eta*dt*clip(G)                # no backward()
    sigma *= sigma_decay
```

8 Further Theoretical Properties

This section separates proved statements from open convergence questions.

8.1 Smooth projected-gradient limit

For a pure diffusion, Theorem 5.2 gives

$$D_{\mathcal{A},\varepsilon}F = (a + \varepsilon I)^{-1} a \nabla F. \tag{59}$$

The regularized operator shrinks sensitivity in weakly excited eigendirections by $\lambda/(\lambda+\varepsilon)$. As $\varepsilon \downarrow 0$, it converges pointwise to $a^{\dagger}a\nabla F$, the orthogonal projection of the ordinary gradient onto the range of the exploration covariance. If a is full rank, the limit is ∇F.

This provides a falsification test: on smooth problems, a sufficiently sampled SBD field should align with the backpropagated gradient in the explored subspace. Persistent disagreement indicates estimator bias, finite-perturbation effects, poor conditioning, or incorrect random-number pairing.

8.2 Expected smooth loss drift

For smooth F and continuous dynamics

$$\mathrm{d}\Theta_t = -A_t G_t \,\mathrm{d}t + \Sigma_t \,\mathrm{d}W_t, \tag{60}$$

Itô's formula yields

$$\mathrm{d}F(\Theta_t) = -\nabla F^{\top} A_t G_t \,\mathrm{d}t + \tfrac{1}{2}\operatorname{tr}(a_t \nabla^2 F)\,\mathrm{d}t + \nabla F^{\top}\Sigma_t \,\mathrm{d}W_t. \tag{61}$$

In the full-rank, small-ε regime $G \approx \nabla F$, the first term is non-positive for $A_t \succeq 0$. The Itô correction can oppose descent at high exploration scale, making annealing structurally necessary.

8.3 Generator field as an L^2 regression

For a finite local move law with displacement $\Delta\psi$ and loss increment ΔF, the regularized field solves

$$D_{\mathcal{A},\varepsilon}F = \arg\min_{g} \left\{ \mathbb{E}_{\mathcal{A}}\left[(\Delta F - g^{\top}\Delta\psi)^2 \mid X_t = x\right] + \varepsilon \|g\|^2 \right\}. \tag{62}$$

Hence $D_{\mathcal{A},\varepsilon}F$ is the best linear predictor of finite loss increments in the stochastic energy geometry induced by the exploration law. In smooth regions it approaches a gradient; across jumps it remains a secant regression.

8.4 Monotonicity of the discrete exponential tilt

The previous manuscript proposed the feedback rule (41) without a descent analysis. The following result supplies the missing justification at the level of the learned regression.

Theorem 8.1 (Predicted-descent monotonicity)**.** *Fix a state x, candidate destinations y, positive base probabilities $p_0(y \mid x)$, and predicted increments $s_y = G(x)^{\top}\delta\psi_{xy}$. Let p_β be defined by Equation (41) and*

$$\mu(\beta) := \mathbb{E}_{p_\beta}[s]. \tag{63}$$

Then

$$\boxed{\frac{\mathrm{d}\mu}{\mathrm{d}\beta} = -\operatorname{Var}_{p_\beta}(s) \le 0.} \tag{64}$$

The inequality is strict whenever the predicted increments are nonconstant on the support of p_β.

Proof. Let $Z(\beta) = \sum_y p_0(y) e^{-\beta s_y}$. Then $\mu(\beta) = -\frac{\mathrm{d}}{\mathrm{d}\beta} \log Z(\beta)$. Differentiating once more gives $\mu'(\beta) = -\frac{\mathrm{d}^2}{\mathrm{d}\beta^2} \log Z(\beta) = -\operatorname{Var}_{p_\beta}(s)$. □

The theorem does *not* say that an inaccurate regression automatically decreases the true loss. It only says that the exponential tilt increasingly favors transitions predicted to reduce loss.

Corollary 8.2 (Sufficient condition for true expected descent)**.** *Suppose every candidate transition obeys a uniform regression-error bound*

$$|\delta F_{xy} - s_y| \le \delta. \tag{65}$$

Then

$$\mathbb{E}_{p_\beta}[\delta F] \le \mu(\beta) + \delta. \tag{66}$$

Consequently, if $\mu(\beta) < -\delta$, the next transition has strictly negative expected true loss increment.

This condition makes the role of β explicit: strong tilting is justified only after the bracket regression is accurate enough that its negative predicted drift dominates model error. Without such accuracy, exploration mixing or a trust-region cap on β is necessary.

8.5 Local integrability and non-integrable losses

The carré-du-champ/predictable-bracket identity is a local statement. Global square integrability of $F(X_t)$ is sufficient but not necessary. For heavy-tailed or potentially non-integrable losses, introduce stopping times τ_n that bound the state, loss, and bracket energy, and work with the stopped Dynkin martingales. The theory then applies on $[0, \tau_n]$ whenever these martingales are square integrable. Passing $n \to \infty$ requires additional uniform-integrability or tightness conditions.

For genuinely infinite-horizon objectives, SBD does not provide a free convergence theorem. Discounted risk requires finiteness of the discounted expectation; average-cost formulations typically require recurrence/ergodicity and a well-defined long-run occupation measure. In practice one may also optimize bounded robust transformations or clipped episodic returns, but that changes the objective and should be stated explicitly.

8.6 Minibatch variation and singular covariance

Equation (58) shows that i.i.d. same-minibatch pairing targets the population cross-moment in expectation. If the exploration covariance is singular, regularization keeps the estimator well-defined:

$$\widehat{G} = (\widehat{Q} + \varepsilon I)^{-1} \widehat{C}. \tag{67}$$

The field is then a ridge regression in the excited subspace; no attempt is made to infer sensitivity in unobserved directions. Rotating or blockwise exploration can cover additional subspaces over time, but under a fixed oracle budget this necessarily reduces the number of observations per subspace.

8.7 Jump-law sensitivity

For a jump kernel with finite second cross-energy, the generator field depends on the proposal law through both

$$\int \gamma\gamma^\top \nu(\mathrm{d}z) \quad \text{and} \quad \int \gamma\, \Delta_\gamma F\, \nu(\mathrm{d}z). \tag{68}$$

Therefore Gaussian, Laplace, Student-t, or structured discrete jumps generally produce different finite-scale secant fields. There is no theorem making Student-t universally optimal; heavy tails simply allocate more probability to long transitions. The empirical ablation confirms task sensitivity.

8.8 Open convergence problem

Conjecture 8.3 (Annealed SBD convergence)**.** *For a coercive locally Lipschitz objective, sufficiently rich rotating exploration, Robbins–Monro-type drift gains, controlled ridge regularization, and an exploration schedule whose late stochastic energy vanishes appropriately, a continuous SBD scheme may converge to a suitable stationary set; under stronger irreducibility and annealing assumptions, a pure-jump discrete scheme may concentrate on global minima.*

We do not claim this as a theorem. The continuous case would require stochastic approximation with biased/noisy bracket regression and nonsmooth differential inclusions. The discrete case would require coupling the evolving regression error to nonhomogeneous Markov-chain annealing. Theorem 8.1 is intentionally local and conditional rather than a global-convergence substitute.

8.9 Dimensionality remains the central obstacle

A scalar loss difference is a narrow channel for identifying a high-dimensional field. Sparse excitation, blockwise clocks, antithetic paths, orthogonal designs, low-rank adapters, distributed perturbations, and learned proposal covariances are potential variance-reduction mechanisms, but none repeal the underlying information constraint. The scaling experiments in Section 9 should therefore be read as an initial stress test, not evidence that SBD is ready for large end-to-end neural training.

9 Empirical Evaluation

The v4 experiments are designed to answer the review's central empirical questions: whether SBD trains real neural classifiers, how it compares with a classical two-query zeroth-order baseline, what happens as parameter dimension and nonsmoothness increase, whether the discrete formulation adds value beyond established search, and whether the jump law matters. All reported tables are generated from checked-in CSV/JSON artifacts in `results/` and `benchmarks/neural/runs_v4/`.

9.1 Evaluation protocol

We report mean test accuracy and sample standard deviation over three seeds for neural benchmarks. For SBD and SPSA, the oracle budget is the number of forward loss evaluations; both use two function evaluations per update. Adam/SGD use one forward/backward training evaluation per update and we separately report gradient evaluations. These costs are not computationally identical, so the tables do not imply that one forward query equals one reverse-mode gradient evaluation; the accounting is intended to make the information budget visible.

The primary neural SBD configuration uses sparse Rademacher innovations, excitation probability $p = 0.02$, excitation-time EMA, and no backpropagation. The benchmark tests monkey-patch `Tensor.backward` and `torch.autograd.grad` to fail inside SBD. The full neural benchmark source is included under `benchmarks/neural/`.

9.2 Standard classification benchmarks

Table 2 reports linear-classifier results on scikit-learn Digits (1,797 handwritten 8×8 images, 650 trainable parameters including bias) and Breast Cancer Wisconsin (62 trainable parameters for the binary linear head in the benchmark encoding).

On both small benchmarks SBD slightly exceeds the matched-query SPSA configuration, which is useful evidence that persistent bracket accumulation can carry information beyond one-step

Table 2: Three-seed standard classification results. SBD/SPSA use no gradient evaluations. Adam/SGD remain substantially more query-efficient on these smooth supervised tasks.

Dataset	Method	Steps	Forward loss queries	Gradient evals	Test accuracy
Digits	SBD-v4	1000	2000	0	$83.33\% \pm 0.83\%$
Digits	SPSA	1000	2000	0	$80.00\% \pm 2.22\%$
Digits	Adam	300	300	300	$95.74\% \pm 0.42\%$
Digits	SGD	300	300	300	$95.93\% \pm 0.32\%$
Breast Cancer	SBD-v4	500	1000	0	$97.08\% \pm 1.34\%$
Breast Cancer	SPSA	500	1000	0	$95.91\% \pm 2.53\%$
Breast Cancer	Adam	200	200	200	$98.25\% \pm 0.88\%$
Breast Cancer	SGD	200	200	200	$97.95\% \pm 0.51\%$

Table 3: Scaling ablation on a 2,410-parameter MLP, three seeds. Tensor cycling lowers instantaneous dimension but also reduces the observation rate of each tensor.

Method	Parameters	Queries	Test accuracy
SBD-all tensors	2410	2000	$72.78\% \pm 1.92\%$
SBD tensor-cycle	2410	2000	$49.07\% \pm 8.91\%$

simultaneous perturbation. However, the gradient methods attain higher accuracy with far fewer optimizer steps. SBD is therefore not competitive with backpropagation on these tasks in its current form.

9.3 Effect of the excitation-time correction

The earlier sparse implementation decayed bracket statistics for coordinates that were not perturbed on a given step. After changing to Equation (55), the seed-0 1,000-step Digits accuracy increases from approximately 66.9% to 83.3% at the same 2,000-query budget. This is an algorithmic correction, not merely hyperparameter tuning: sparse unexcited coordinates are missing observations and should not be interpreted as zero cross-covariation samples.

9.4 Scaling and blockwise ablation

To test whether structured excitation automatically solves high-dimensional variance, we train a 2,410-parameter one-hidden-layer nonlinear MLP on Digits with either all tensors eligible for sparse excitation or one tensor excited per step in a deterministic cycle. Both receive 2,000 forward queries.

The negative blockwise result is informative. Reducing simultaneous dimensionality is not enough; the query budget must also compensate for the slower observation clock of each block. Future low-rank/blockwise variants should therefore report effective observations per subspace rather than only the total number of optimizer steps.

9.5 Nonsmooth ReLU architecture

A 1,210-parameter ReLU MLP on Digits reaches $51.85\% \pm 5.61\%$ after 2,000 SBD forward queries and zero gradient evaluations. ReLU itself is not a fundamental obstacle because paired finite loss increments are defined across activation kinks. The result nevertheless shows that end-to-end nonlinear neural optimization is currently much harder for the diagonal SBD estimator than the small linear benchmarks.

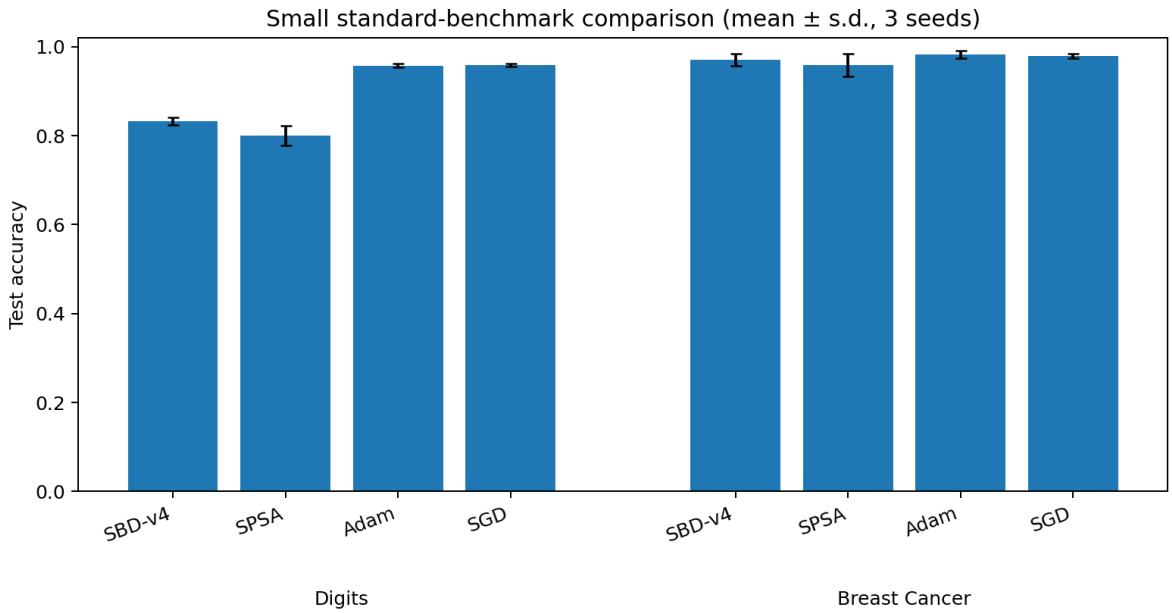

Figure 4: Mean test accuracy with one-standard-deviation error bars over three seeds. The principal empirical conclusion is feasibility without gradients, not parity with backpropagation.

Table 4: Discrete 24-bit Hamming task, 50 seeds, 300-step/evaluation budget. The task is separable and therefore too easy to distinguish several competent discrete optimizers.

Method	Mean best loss	Exact-solution rate	Mean final loss
Generator-SBD	0.00	1.00	0.54
Random walk	5.94	0.00	12.26
Greedy hill climb	0.00	1.00	0.00
Simulated annealing	0.00	1.00	0.00
Cross-entropy method	0.00	1.00	0.04

9.6 Pure-jump discrete baselines

The v3 binary experiment optimizes 24-bit Hamming loss from the all-zero state over 50 seeds/targets and a budget of 300 proposals or evaluations. Table 4 adds the requested discrete baselines.

The original comparison against random walk correctly showed that the learned generator tilt carries transition information. It did *not* establish an advantage over established discrete stochastic optimization. Hill climbing, annealing, and CEM all solve this simple separable problem. A stronger publication benchmark should use deceptive/epistatic binary landscapes, constrained architecture graphs, discrete routers, or combinatorial objectives in which local one-bit descent is insufficient.

9.7 Generator mechanism checks

The exact finite generator implementation is also tested independently. On a three-dimensional quadratic with symmetric coordinate moves of scale 10^{-4}, the finite generator-bracket field has ℓ_2 error 2.34×10^{-10} relative to the analytic gradient and cosine alignment numerically equal to one. On a piecewise-constant two-dimensional loss, a finite pure-jump generator produces a nonzero field

Table 5: Jump-law sensitivity. No law dominates across both tasks; Student-t is therefore an optional proposal choice rather than part of the SBD definition.

Task	Jump law	Mean best loss	Zero-loss rate	Loss < 0.02 rate
Plateau	Gaussian	1.100	0.533	0.533
Plateau	Laplace	1.067	0.300	0.300
Plateau	Student-t	1.533	0.233	0.233
Hard threshold	Gaussian	0.216	0.167	0.267
Hard threshold	Laplace	0.162	0.200	0.467
Hard threshold	Student-t	0.188	0.233	0.467

of norm $\sqrt{2} \approx 1.414$ even though infinitesimal derivatives provide no useful local signal. These are implementation/mechanism checks rather than competitive optimization benchmarks.

9.8 Jump-law ablation

Table 5 compares Gaussian, variance-normalized Laplace, and Student-t compound-Poisson amplitudes over 30 seeds on the annular plateau and hard-threshold toy problems. All other settings are held fixed.

Gaussian jumps work best on the plateau under this configuration, while Laplace and Student-t achieve more very-low-loss runs on the hard threshold. The relevant design variable is the scale and geometry of nonlocal exploration, not a universally optimal heavy-tailed family.

9.9 What these experiments do and do not establish

The experiments establish four limited claims. First, SBD can train standard neural classifiers without gradient evaluations. Second, the excitation-time estimator is materially better than the previous sparse EMA. Third, a matched-query SPSA baseline does not trivially dominate SBD on the two small linear benchmarks. Fourth, the generator formulation behaves as predicted on discontinuous and discrete objectives.

They do not establish large-model scalability, wall-clock competitiveness, state-of-the-art derivative-free performance, or global convergence. MNIST support remains in the included neural benchmark repository, but this revision does not report a multi-seed MNIST result because the current verified study focused on fully reproducible local datasets. CIFAR-scale tests remain future work.

10 Limitations, Failure Modes, and Research Risks

SBD has several serious limitations that remain open after the v4 refactor.

10.1 Forward-query and wall-clock efficiency

The reference SBD step uses two forward loss evaluations. Reverse-mode differentiation often costs only a small multiple of one forward pass while returning sensitivity for all parameters simultaneously. The benchmark confirms this disadvantage: Adam/SGD reach higher Digits accuracy in far fewer optimization steps than SBD. The present evidence therefore supports SBD mainly where gradients are unavailable, artificial, or expensive to expose, not as a general replacement for backpropagation.

10.2 High-dimensional variance

A scalar loss difference is a low-bandwidth observation of a high-dimensional field. Sparse excitation and excitation-time averaging improve the estimator, but the 2,410-parameter MLP experiment remains substantially weaker than the linear case. Tensor cycling performs worse under a fixed query budget because each tensor receives fewer observations. Scaling requires structured subspaces, distributed perturbations, low-rank adapters, orthogonal designs, or other variance-reduction mechanisms that are not yet validated at modern deep-learning scale.

10.3 Regularization does not create information

The ridge parameter ε guarantees a well-defined inverse even when the empirical bracket matrix is singular, but it cannot infer unexcited directions. In the smooth limit, the field is a shrunk projection onto the explored range. Rotating subspaces can improve coverage only by spending additional oracle budget.

10.4 The jump-feedback guarantee is conditional

Theorem 8.1 proves monotonicity of the *predicted* transition increment under exponential tilting. Corollary 8.2 requires the regression error to be smaller than the negative predicted drift. There is no unconditional theorem that an inaccurate bracket field makes a discrete optimizer descend, and no global-convergence theorem is claimed for the adaptive nonhomogeneous Markov chain.

10.5 Discrete baselines weaken the original toy claim

Generator-SBD dominates an unbiased random walk on the 24-bit Hamming problem, but greedy hill climbing, simulated annealing, and the cross-entropy method also solve it in every tested seed. The example therefore validates information transfer into the generator tilt but does not demonstrate an optimization advantage. Harder epistatic or constrained discrete benchmarks are required.

10.6 Nonsmooth and ReLU learning is feasible but inefficient

SBD does not require differentiability through ReLU kinks, hard thresholds, quantizers, or discrete transitions because finite paired loss differences remain defined. Nevertheless, the ReLU benchmark reaches only $51.85\% \pm 5.61\%$ on Digits under the tested budget. "Defined" and "practically efficient" are distinct claims.

10.7 Data and simulator noise

Common random numbers reduce nuisance variance and are unbiased for the population cross-moment when minibatch/simulator randomness is exogenous and independent of the innovation. If proposal-dependent sampling, adaptive data selection, nonstationarity, or simulator state creates dependence, Equation (58) no longer applies directly and the bracket target may shift. Such settings require explicit modeling or correction.

10.8 Integrability and infinite horizon

The generator theory is local. Non-integrable heavy-tailed losses require localization, truncation, or additional moment conditions. Infinite-horizon discounted or average-cost objectives require

their own recurrence/ergodicity assumptions. The manuscript does not claim that any arbitrary measurable long-horizon loss automatically admits the displayed carré-du-champ identities.

10.9 Exploration-law sensitivity

The jump-law ablation shows no universal winner among Gaussian, Laplace, and Student-t proposals. Heavy tails can help cross discontinuities but can also overshoot useful regions. Brownian scale, jump rate, tail law, and annealing schedule are part of the optimizer design and may be problem dependent.

10.10 Novelty is a combination claim

Quadratic covariation, carré-du-champ operators, Dirichlet forms, SPSA, stochastic approximation, Lévy jumps, simulated annealing, cross-entropy search, and derivative-free neural optimization all have substantial prior literatures. The proposed novelty is specifically the normalized state–loss generator energy as a common optimizer field across continuous diffusion, finite jumps, and pure-jump discrete trainable states, together with its feedback use. Discovery of prior work implementing this exact construction would narrow the claim and should be incorporated transparently.

10.11 Deployment safety

A jump-based optimizer can make abrupt parameter changes. Any online or consequential deployment would require projection constraints, validation gates, trust regions, rollback checkpoints, and independent monitoring. The present code is research software, not a production optimizer.

11 Conclusion

Semimartingale Bracket Descent proposes that stochastic state motion can supply the calculus of learning rather than merely perturb a pre-existing gradient method. Its central regularized operator is

$$\boxed{D_{\mathcal{A},\varepsilon}F = \big(\Gamma_{\mathcal{A}}(\psi,\psi) + \varepsilon I\big)^{-1}\Gamma_{\mathcal{A}}(\psi,F), \qquad \varepsilon > 0.} \tag{69}$$

For continuous diffusions this becomes a ridge-shrunk gradient in the excited geometry and converges to the projected ordinary gradient as $\varepsilon \downarrow 0$. For finite jumps it is a nonlocal secant regression. For finite/countable parameter spaces it is a graph-space regression over Markov transitions, so the learning law remains meaningful when no differentiable parameter trajectory exists.

The v4 revision strengthens both theory and evidence. Generator-domain and local-integrability assumptions are explicit; the unnecessary constant-rank requirement is removed from the regularized optimizer; exponential jump tilting receives a predicted-descent theorem and a regression-error condition for true expected descent; and sparse neural SBD now uses excitation-time rather than global-step bracket averaging.

The empirical picture is mixed and therefore informative. On small standard classification tasks, SBD learns without any gradient evaluations and slightly exceeds a matched-query SPSA configuration, but Adam/SGD remain much more efficient. A larger nonlinear MLP and ReLU ablation expose substantial scaling difficulty. In the discrete setting, Generator-SBD clearly improves over random walk but does not beat hill climbing, simulated annealing, or the cross-entropy method on the simple separable Hamming task. The jump-law ablation further shows that Student-t is a proposal choice rather than an intrinsic component.

The strongest remaining research question is consequently narrow and testable: can normalized stochastic state–loss energy become an information-efficient optimizer on problems where differentiation is unavailable, structurally inappropriate, or confined to only part of a hybrid architecture? The present manuscript establishes a unified calculus, an executable estimator, a local feedback guarantee, and initial benchmark evidence; it does not yet establish large-scale superiority.

A Derivations and Extensions

A.1 Full matrix bracket estimator

Given innovations $m_1, \dots, m_n \in \mathbb{R}^d$ and paired loss increments Δy_k, define

$$Q_n = \sum_{k=1}^{n} w_k m_k m_k^\top, \qquad C_n = \sum_{k=1}^{n} w_k m_k \Delta y_k. \tag{70}$$

The ridge-regularized estimator

$$\widehat{G}_n = (Q_n + \varepsilon I)^{-1} C_n \tag{71}$$

solves

$$\widehat{G}_n = \arg\min_g \sum_{k=1}^{n} w_k (\Delta y_k - g^\top m_k)^2 + \varepsilon \|g\|^2. \tag{72}$$

Thus bracket descent admits a direct statistical interpretation as online local regression of loss increments onto martingale increments.

A.2 Random-subspace estimator

Let $P \in \mathbb{R}^{d \times r}$ have orthonormal columns and $m = Pz$. Then

$$Q_z = \sum z_k z_k^\top, \qquad C_z = \sum z_k \Delta y_k, \tag{73}$$

and the parameter-space field is

$$\widehat{G} = P(Q_z + \varepsilon I_r)^{-1} C_z. \tag{74}$$

Memory drops from $O(d^2)$ to $O(dr + r^2)$ if P is explicit, or lower when P is generated procedurally from a seed.

A.3 Orthogonal innovations

Instead of independent Gaussian vectors, one can use randomized orthogonal or Hadamard directions within a block. If $z_1, \dots, z_r$ form an orthogonal basis with equal norm, then Q_z is proportional to the identity over a complete local sweep, reducing conditioning noise. Random signs preserve an approximately Brownian interpretation at the coarse-grained level while improving finite-sample coverage.

A.4 Antithetic bracket estimator

A variance-reduced variant evaluates $F(\theta + m)$ and $F(\theta - m)$ on the same minibatch and uses

$$\Delta y^{\pm} = \frac{F(\theta + m) - F(\theta - m)}{2}. \tag{75}$$

Then $m\Delta y^{\pm}$ cancels even-order perturbation terms for smooth F. This costs an additional forward evaluation and moves SBD closer to classical two-sided zeroth-order estimators, so the one-sided pathwise version remains the conceptually pure reference algorithm.

A.5 Hybrid differentiable/non-differentiable modules

Partition parameters as (θ_D, θ_N) where θ_D belongs to differentiable modules and θ_N to non-differentiable modules. A hybrid step is

$$\theta_D^{k+1} = \theta_D^k - \eta_D \widehat{\nabla}_{\theta_D} F, \tag{76}$$

$$\theta_N^{k+1} = \theta_N^k + m_k - \eta_N h \widehat{G}_k^{\mathrm{SBD}}. \tag{77}$$

This is likely to be the most compute-efficient use of SBD in large networks. It resembles earlier hybrid ES ideas in spirit, but the non-differentiable component is trained with bracket statistics and can include an explicit jump channel.

A.6 State-dependent exploration

One may let $\Sigma(\theta)$ depend on the current parameter state. In the smooth regime, Theorem 4.2 still gives

$$\mathrm{d}\left\langle M^{\Theta}, M^{Y}\right\rangle = a(\theta)\nabla F(\theta)\mathrm{d}t, \tag{78}$$

so bracket normalization removes the first-order anisotropy provided a is identifiable and sufficiently excited. This suggests adaptive exploration that allocates more stochastic energy to uncertain or underexplored parameter blocks without changing the target field in the full-rank smooth limit.

A.7 Constrained parameters

For box, simplex, orthogonality, or manifold constraints, one may project the post-step state or define the stochastic motion intrinsically on the constraint set. On a Riemannian manifold $\mathcal{M}$, a natural research direction is to replace Euclidean quadratic variation with the covariation of tangent-space martingales and use stochastic development to define a manifold-valued SBD.

A.8 Connection to generalized stochastic derivatives

For square-integrable martingales, Kunita–Watanabe theory decomposes one martingale into a component stochastic-integrable with respect to another plus an orthogonal martingale. In schematic vector form,

$$M_t^Y = \int_0^t G_s^{\top} \mathrm{d}M_s^{\Theta} + M_t^{\perp}. \tag{79}$$

The bracket identity identifies G from predictable covariation. SBD can therefore be viewed as converting the integrand in a martingale representation into a control drift for the parameter process. This interpretation may be a useful bridge to filtering, stochastic control, and online system identification.

A.9 Derivation of the generator-bracket identity

For an Itô jump diffusion with coordinate embedding $\psi(x) = x$, let $\mathcal{A}$ be the extended generator and define $\Gamma_{\mathcal{A}}(f, g) = \mathcal{A}(fg) - f\mathcal{A}g - g\mathcal{A}f$. Applying the product rule to the diffusion component gives $\nabla f^{\top} a \nabla g$. The jump component contributes

$$\int [f(x+\gamma) - f(x)][g(x+\gamma) - g(x)]\, \nu_x(\mathrm{d}z). \tag{80}$$

Taking $f = \psi_i$ and $g = F$ yields

$$\Gamma_{\mathcal{A}}(\psi, F) = a\nabla F + \int \gamma\, \Delta_\gamma F\, \nu(\mathrm{d}z), \tag{81}$$

while $f = \psi_i, g = \psi_j$ yields

$$\Gamma_{\mathcal{A}}(\psi, \psi) = a + \int \gamma\gamma^\top \nu(\mathrm{d}z). \tag{82}$$

The predictable compensator of the realized quadratic covariation has this same density. Hence the pathwise bracket regression in the original SBD construction is an online estimator of the generator-normalized field in Equation (32).

A.10 Finite-state generator

For $\mathcal{A}f(x) = \sum_{y \neq x} q(x, y)[f(y) - f(x)]$, substitution into the definition of $\Gamma_{\mathcal{A}}$ gives

$$\Gamma_{\mathcal{A}}(f, g)(x) = \sum_{y \neq x} q(x, y)[f(y) - f(x)][g(y) - g(x)]. \tag{83}$$

This identity requires no differentiable structure on E and directly yields Proposition 5.3.

B Reproducibility and Repository Guide

The repository is organized so that the mathematical mechanism tests and neural benchmarks can be reproduced independently.

B.1 Environment

The root package requires Python 3.10+, NumPy, PyTorch, and pytest. The integrated neural benchmark additionally uses scikit-learn, matplotlib, and torchvision. A convenience installation is

```
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
pytest -q
```

The root pytest configuration uses importlib mode to avoid module-name collisions between the root tests and the embedded neural benchmark tests.

B.2 Mechanism experiments

The continuous/nonsmooth toy suite is reproduced with

```
python experiments/run_toys.py
python experiments/run_generator_toys.py
python experiments/run_jump_law_ablation.py
```

These commands regenerate the CSV files under `results/` for the smooth generator identity, discontinuous jump field, discrete baseline comparison, and jump-law sensitivity study.

B.3 Neural benchmark

The integrated benchmark lives under `benchmarks/neural/`. It contains SBD, SPSA, Adam, and SGD trainers, standard data loaders, models, JSON configurations, tests, and checked-in per-seed v4 runs. The SBD path intentionally contains no call to `loss.backward()` or `torch.autograd.grad()`.

The small offline Digits benchmark can be run after installing the benchmark package:

```
cd benchmarks/neural
pip install -e ".[dev]"
python -m sbd_bench.train --dataset digits --model linear \
  --optimizer sbd --full-batch --max-steps 1000 \
  --sbd-element-prob 0.02 --output runs/digits_sbd.csv
```

SPSA uses the same two-forward-query accounting. Adam/SGD report gradient evaluations separately. MNIST code and configurations are included, but the v4 manuscript reports only the verified local-data multi-seed experiments.

B.4 Checked-in result files

The principal manuscript tables are backed by:

- `results/neural_benchmark_summary.csv`;
- `results/generator_graph_summary.csv`;
- `results/generator_mechanism.csv`;
- `results/jump_law_ablation_summary.csv`;
- `benchmarks/neural/runs_v4/`, which stores individual neural runs.

A SHA-256 manifest is regenerated for the final archive.

B.5 Interpretation rule

The repository deliberately preserves negative results. The nonlinear/blockwise SBD ablation is weaker than the linear benchmark; the ReLU run is far from conventional neural performance; and the discrete Hamming baseline does not distinguish Generator-SBD from established search. Reproduction should therefore compare both successes and failures rather than only the headline SBD runs.

References

[1] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. *ICML*, 2011.

[2] M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient Langevin dynamics: a nonasymptotic analysis. *COLT / PMLR 65*, 2017.

[3] E. Naldi, M. Rando, L. Rosasco, and S. Villa. Langevin for nonconvex optimization: exact, inexact and zeroth-order. *arXiv:2607.22353*, 2026.

[4] H. Li and J. C. Spall. Zeroth-order Langevin Monte Carlo via SPSA under noisy function measurements. *arXiv:2608.07837*, 2026.

[5] J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992. DOI: 10.1109/9.119632.

[6] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. DOI: 10.1126/science.220.4598.671.

[7] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004. DOI: 10.1007/978-1-4757-4321-0.

[8] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv:1703.03864*, 2017.

[9] K. Lenc, E. Elsen, T. Schaul, and K. Simonyan. Non-differentiable supervised learning with evolution strategies and hybrid methods. *arXiv:1906.03139*, 2019.

[10] E. Gorbunov, P. Dvurechensky, and A. Gasnikov. An accelerated method for derivative-free smooth stochastic convex optimization. *SIAM Journal on Optimization*, 2022.

[11] A. Bibi, E. H. Bergou, O. Sener, B. Ghanem, and P. Richtárik. A stochastic derivative-free optimization method with importance sampling: theory and learning to control. *AAAI*, 2020.

[12] R. Bollapragada, C. Karamanli, and S. M. Wild. Derivative-free stochastic optimization via adaptive sampling strategies. *Optimization Methods and Software*, 2025.

[13] B. Engquist, K. Ren, and Y. Yang. Adaptive state-dependent diffusion for derivative-free optimization. *Communications on Applied Mathematics and Computation*, 2024.

[14] M. Fornasier, H. Huang, L. Pareschi, and P. Sünnen. Anisotropic diffusion in consensus-based optimization on the sphere. *SIAM Journal on Optimization*, 2022.

[15] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11:3137–3181, 2010.

[16] Q. Rong, L. Yan, X. Duan, J. Liu, and Y. Jia. Optimal control framework and theoretical convergence guarantees for jump-diffusion stochastic neural networks. *Neurocomputing*, 700:134427, 2026.

[17] Z. Guo, W. Tang, and R. Xu. Conditional diffusion guidance under hard constraint: a stochastic analysis approach. *arXiv:2602.05533*, 2026.

[18] S. A. Almada Monter. Quadratic covariation estimates in non-smooth stochastic calculus. *Stochastic Processes and their Applications*, 125(1):343–361, 2015.

[19] D. Filipović and M. Larsson. Polynomial jump-diffusion models. *Stochastic Systems*, 10(1):71–97, 2020.

[20] N. Bouleau. Differential calculus for Dirichlet forms: the measure-valued gradient preserved by image. *Journal of Functional Analysis*, 225(1):63–73, 2005.

[21] J. Bamberger, I. Jones, D. Duncan, M. M. Bronstein, P. Vandergheynst, and A. Gosztolai. Carré du champ flow matching: better quality–generalisation tradeoff in generative models. *arXiv:2510.05930v2*, 2026.

[22] P. E. Protter. *Stochastic Integration and Differential Equations*. Springer, 2nd ed., 2005.

[23] D. Revuz and M. Yor. *Continuous Martingales and Brownian Motion*. Springer, 3rd ed., 1999.

[24] B. Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer, 6th ed., 2003.

[25] F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.