NOISE IN DIFFUSION MODELS IS A LEARNABLE INPUT

Anonymous authors
Paper under double-blind review

ABSTRACT

Model training consumes concrete random inputs that enter the realized loss and optimization. Because training aims to reduce this loss, models can learn to exploit accessible structure in these inputs, making realized randomness itself learnable content. We establish this learning mechanism in diffusion models, where clean data and target noise jointly form the noisy input. Prediction gains can arise from clean-data regularities, noise structure, and their interaction, so lower loss need not indicate better learning of clean-data regularities. We test this mechanism with controlled, reproducible pseudorandom number generator (PRNG) streams on MNIST and CIFAR-10. Random-role ablations localize the dominant effect to diffusion noise. Without reusable clean-image structure, structured-noise training can lower loss below the IID reference, while IID testing reverses this advantage, demonstrating learned noise dependence. Value-preserving shuffling largely removes these gains, identifying ordering as a major source of learnable structure. Matched-source initialization improves representative generations, supporting learned noise as a generation cue. Prior results on noise–data assignment, noise-based backdoors, and structured video noise provide independent evidence of learned use of noise structure and noise–data relationships, including in large generative models. These studies also demonstrate that noise design can exploit this learnability to achieve specific goals or improve model performance. These findings motivate testing whether existing large generative models learn structure in industrial-grade PRNG outputs that affects generation quality. Our view establishes noise as a learnable and therefore designable input dimension for guiding learning and generation.

1 INTRODUCTION

Random inputs are integral to modern learning systems. They appear, for example, as random masks in dropout (Srivastava et al., 2014), reparameterization noise in variational autoencoders (Kingma & Welling, 2014), and the noise used in diffusion training (Ho et al., 2020). Yet stochastic learning is usually described in terms of abstract random variables and distributions, whereas actual training repeatedly consumes concrete realized values. These inputs enter the realized loss and optimization. Because training aims to reduce this loss, models can learn to exploit accessible structure in these inputs. Our central question is therefore whether and how structure entering through random inputs can become learnable content, rather than merely a source of stochastic variation. We develop this view generally and study its concrete mechanism in diffusion models.

Prior work has shown that the choice and organization of noise can substantially affect diffusion learning and generation. Non-Gaussian and non-isotropic diffusion alter the underlying noise distributions (Nachmani et al., 2021; Voleti et al., 2022), while blue-noise diffusion introduces structured correlations (Huang et al., 2024). Other methods modify noise–data assignment to accelerate training (Li et al., 2024), introduce noise-based triggers for targeted behavior (Chen et al., 2023), or encode motion through structured noise in video diffusion (Burgert et al., 2025). Together, these results indicate that models can exploit structure introduced through noise. To our knowledge, we are the first to explicitly formulate realized diffusion noise itself as learnable input content and to isolate this general mechanism in diffusion training through controlled experiments.

In noise-prediction diffusion models, this mechanism has a direct form. The clean component and the realized noise jointly form the noisy input, and the target noise is already embedded in that input. The model can reduce prediction error by learning regularities of the clean data, by exploiting

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

structure introduced through the realized noise, or by using both jointly. These two routes can interact because both components enter the same input. Consequently, a lower noise-prediction loss need not indicate better learning of clean-data regularities.

We test this mechanism using controlled pseudorandom streams on MNIST and CIFAR-10. Random-role ablations localize the dominant effect to diffusion noise. After replacing real images with fresh IID tensors, structured-noise training can lower loss below the IID reference, while IID-noise testing raises it above the corresponding test reference. IID-trained models obtain no comparable gain when structured noise is introduced only at test time. These comparisons support learned noise dependence without reusable clean-image structure. Shuffling the same values largely removes the gains while preserving their empirical marginal distribution. This learned dependence also affects generation. These findings raise an important question: whether existing large generative models already learn structure in industrial-grade PRNG outputs, and whether this affects generation quality.

Prior studies provide further evidence of noise learnability, including in large generative models: changing how noise is paired with images can accelerate training; training with noise triggers teaches models to produce chosen outputs; and fine-tuning on temporally correlated noise enables effective video motion control. Our learning view also offers a unified interpretation of these findings: noise supplies information that models can learn to use together with clean-data information. This suggests treating noise as a designable input, with useful structure deliberately introduced or preserved to guide learning and generation.

2 Noise as a Learnable Input

2.1 Realized Objectives under Concrete Random Inputs

A stochastic learning objective is usually written as an expectation over random variables,

$$\mathcal{L}(\theta; D) = \mathbb{E}_R \left[\widehat{\mathcal{L}}(\theta; D, R) \right], \tag{1}$$

where D is the training data, θ denotes the model parameters, and R collects the random variables used during training.

In practice, training does not evaluate this expectation directly. At optimization step k, it consumes concrete values R_k, which may be pseudorandomly generated or otherwise constructed. The realized loss is $\widehat{\mathcal{L}}_k(\theta_k; D, R_k)$, and a gradient update has the form

$$\theta_{k+1} = \theta_k - \eta_k \nabla_\theta \widehat{\mathcal{L}}_k(\theta_k; D, R_k). \tag{2}$$

Thus, the concrete random inputs enter both the realized loss and its gradient. If these inputs contain structure accessible to the model, optimization can learn to exploit it to reduce the loss. The experiments below test whether such structure is actually learned and exploited.

2.2 Diffusion-Specific Mechanism

Diffusion noise is a natural focus for examining this learning view. During standard diffusion training, repeatedly sampling noise tensors typically consumes far more random values than model initialization, timestep sampling, and data shuffling combined. In noise-prediction models, these tensors are embedded in the noisy input and also serve as the prediction target, so learning to exploit their accessible structure can directly reduce prediction error. Specifically, the model receives

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon \tag{3}$$

and predicts ϵ from (x_t, t). Better identification of the clean component can improve noise prediction; accessible structure in the noise can also help predict the same target. These routes interact because both components form the same observation. Under squared loss, the ideal conditional prediction rule for a noise source s is $f_s^*(x_t, t) = \mathbb{E}_s[\epsilon \mid x_t, t]$.

To formalize the prediction opportunity supplied by information beyond one coordinate, define, under a fixed source and clean-data distribution,

$$m_i = \mathbb{E}[\epsilon_i \mid x_{t,i}, t], \qquad M_i = \mathbb{E}[\epsilon_i \mid x_t, t]. \tag{4}$$

The first predictor uses the corresponding noisy scalar and timestep; the second uses the whole noisy tensor. Orthogonality of conditional expectation gives

$$\mathbb{E}[(\epsilon_i - m_i)^2] - \mathbb{E}[(\epsilon_i - M_i)^2] = \mathbb{E}[(M_i - m_i)^2] \geq 0. \tag{5}$$

For real images, this gain may include information from clean-image structure as well as noise dependence. In the ideal IID-clean probe, with clean data independent of the noise, reusable clean-side cross-coordinate structure is absent. If the noise is also IID, other coordinates provide no additional prediction information. Dependent noise can make them informative. Equation 5 describes available prediction information; whether a trained model exploits it is tested in Section 4.

This mechanism is distinct from explicit pseudorandom-sequence prediction (Tao et al., 2025): the diffusion model estimates realized noise already present in its current input rather than predicting a future sequence value.

These considerations motivate four testable expectations: the main source-dependent effects should persist when the tested source supplies only training and evaluation diffusion noise, but weaken when it supplies only the auxiliary random roles; noise-dependent gains may persist after reusable clean structure is removed; changing the test source may remove an advantage learned during training; and shuffling the same values should weaken gains that rely on their ordering. These expectations concern accessible structure and need not have equal strength for every source or finite trajectory.

3 Experimental Setup

We use 13 sources: Logistic, Markov-type, ShiftedSine, and GLIBC streams, together with NumPy's `default_rng` as an operational IID reference. Source definitions are in Appendix A.1. The empirical adequacy of the IID reference is assessed in Section 4.2.

Each non-IID source is transformed pointwise by a fixed empirical-rank map followed by the inverse-normal CDF and separately seeded mean/variance calibration. The transform does not reorder the stream. The empirical adequacy of this marginal control for the present task is assessed in Section 4.2; the construction is specified in Appendix A.1.

Unless stated otherwise, the tested source supplies training and evaluation diffusion noise, while initialization, timestep sampling, data shuffling, and standard sampling use IID. Role seeds are matched across sources. Training and evaluation noise use separately seeded trajectories. Full-PRNG and Auxiliary-only routing are used for Section 4.1; the routing and seed protocol are detailed in Appendix A.2.

We train on MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky, 2009). The IID-clean probe retains the corresponding U-Net, forward process, optimizer, and epsilon-prediction objective (Ho et al., 2020), but replaces real clean inputs with fresh online IID tensors uniform on $[-1, 1]$. Models use 1,000 diffusion timesteps. Evaluation reuses a fixed bank of 4,096 clean examples, timesteps, and noise tensors, and test tail loss averages the final 10 evaluations. Standard generation uses 100-step deterministic DDIM (Song et al., 2021) with $\eta = 0$ and IID initial noise. Training budgets, architecture, optimization, and sampling details are in Appendix A.2.

We exclude absorbing-state failures and PRBS31 period reuse under the long training budget; Appendix B documents these distinct cases. Main comparisons use run 1; additional run-2 and run-3 role and probe plots are provided separately in Appendix E.

4 Mechanistic Evidence for Noise Learnability

We examine five aspects of noise learnability: which random roles carry the effect, whether learning persists without reusable clean-image structure, which noise properties support it, whether learned gains transfer across finite trajectories, and how this dependence affects generation. The generation differences are large—from disordered textures to recognizable digits. We therefore present representative samples rather than automated quality metrics, since the images convey these changes more directly.

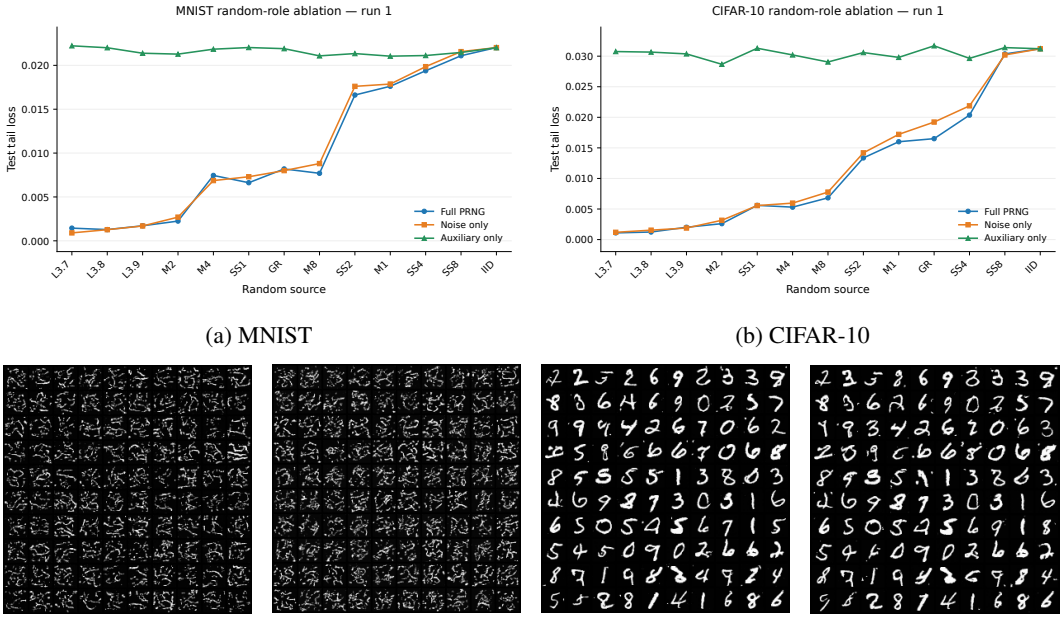
4.1 Localizing the Learnable Signal to Diffusion Noise

We separate noise roles (training and evaluation diffusion noise) from auxiliary roles (model initialization, timestep sampling, and data shuffling). Full PRNG uses the tested source for both groups; Noise-only and Auxiliary-only use it only for the corresponding group, with IID supplying the other group. All generation comparisons use IID initial noise and deterministic DDIM with $\eta = 0$.

Following Section 2.2, we expect the dominant source-dependent effect to arise from diffusion noise. The test losses on both datasets support this expectation (Figure 1(a,b)):

$$L_{\text{NoiseOnly}} \approx L_{\text{FullPRNG}}, \qquad L_{\text{AuxiliaryOnly}} \approx L_{\text{IID}}.$$

The generation results on MNIST follow the same grouping (Figure 1(c–f)): Full PRNG and Noise-only show severe degradation despite their lower prediction losses, whereas Auxiliary-only remains close to IID. Thus, both loss and generation comparisons localize the dominant effect to diffusion noise.



(a) MNIST (b) CIFAR-10

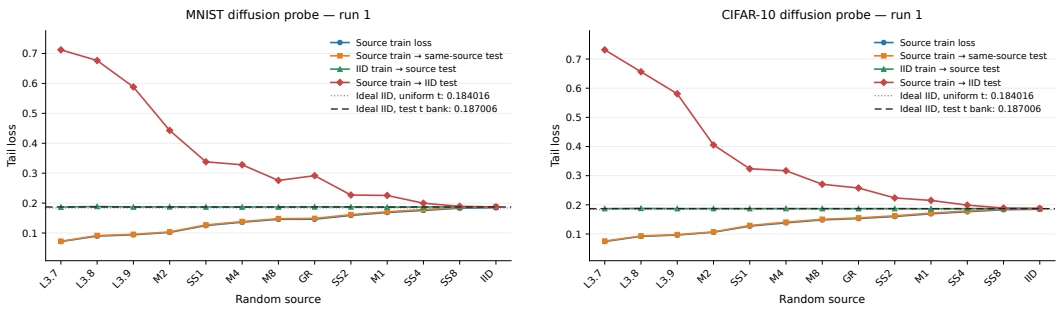
(c) Full PRNG (d) Noise-only (e) Auxiliary-only (f) IID

Figure 1: Random-role ablation. Top: run-1 test tail loss on MNIST and CIFAR-10. Noise-only closely follows Full PRNG across random sources, whereas Auxiliary-only remains near the IID level. Bottom: representative MNIST generations for GR under Full PRNG, Noise-only, Auxiliary-only, and IID. Full PRNG and Noise-only show similarly severe degradation, whereas Auxiliary-only remains close to IID in generation quality.

4.2 Removing Reusable Clean-Side Structure

To remove reusable clean-image structure, we replace real images with fresh online IID tensors uniform on $[-1, 1]$, retaining the noise-prediction task. Under ideal IID Gaussian noise, let L_0 denote the minimum expected MSE under uniformly sampled training timesteps, and $L_{0,\text{bank}}^{(r)}$ the corresponding reference for run r's fixed test timesteps. Writing L_t^* for the minimum expected MSE at timestep t, Appendix C gives

$$\begin{aligned} L_0 &= \frac{1}{T}\sum_{t=0}^{T-1} L_t^* \approx 0.184016, \\ L_{0,\text{bank}}^{(r)} &= \frac{1}{N}\sum_{i=1}^{N} L_{t_i^{(r)}}^*, \qquad N = 4096. \end{aligned} \tag{6}$$



(a) MNIST (b) CIFAR-10

Figure 2: Diffusion-probe train–test mismatch on MNIST and CIFAR-10 using run 1. The dotted line is the ideal IID reference for uniformly weighted timesteps ($L_0 = 0.184016$); the dashed line uses the fixed run-1 test timesteps ($L^{(1)}_{0,\text{bank}} = 0.187006$). Training with structured sources can reduce loss below the uniform-timestep reference, whereas source-trained/IID-tested losses lie above or near the test reference. IID-trained/source-tested losses remain near the test reference. Same-source test losses are displayed separately; their relation to training loss depends on transfer across separately seeded trajectories.

Measured IID-to-IID test losses closely match the run-specific reference, supporting the operational IID reference in Section 3 (Appendix C, Table 1).

Let $L_{\text{train}}(s)$ denote training loss with noise source s, and $L_{a\to b}$ test loss after training with a and testing with b. Following Section 2.2, we expect learning accessible noise structure to reduce training loss even without reusable clean-image structure. A model relying on these regularities should lose or reverse this advantage under IID testing. This motivates the predictions

$$L_{\text{train}}(s) \lesssim L_0, \qquad L_{s\to\text{IID}} \gtrsim L^{(r)}_{0,\text{bank}}.$$

When the learned noise dependence is dominated by regularities shared across trajectories of the same PRNG, we also expect $L_{s\to s} \approx L_{\text{train}}(s)$.

Figure 2 supports the two core predictions on both datasets: training losses fall below or near the training reference, whereas source-trained/IID-tested losses lie above or near the test reference. Same-source test losses remain close to training losses for many sources, while larger gaps also occur. Section 4.4 examines how learning trajectory-specific regularities can account for such differences.

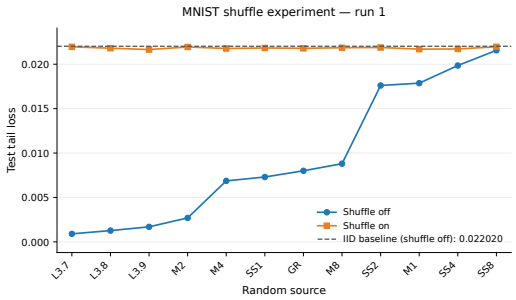
Conversely, we introduce structured noise only at test time to IID-trained models, without further training or parameter updates. Figure 2 shows $L_{\text{IID}\to s} \approx L^{(r)}_{0,\text{bank}}$: changing the test noise alone does not reproduce the gains obtained through source-specific training. Agreement with the IID reference across Gaussianized sources also supports the empirical adequacy of the marginal control used in Section 3 for this task.

4.3 Destroying Dependence While Preserving Marginals

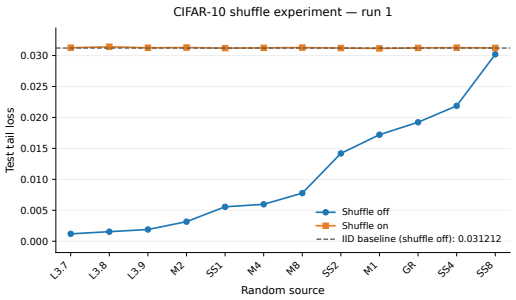
The empirical support for marginal control in Section 4.2 suggests that the learned gains arise mainly from the ordering of noise values rather than their scalar distribution. Following Section 2.2, randomizing this ordering should disrupt the exploitable dependencies and largely remove the associated prediction advantage. We therefore expect

$$L_{\text{shuffle}} \approx L_{\text{IID}}.$$

We globally shuffle each source's training and evaluation streams separately using independently generated permutations, preserving exactly the same values and empirical marginals within each stream (Appendix A.3). Figure 3 supports the prediction on both datasets: the source-dependent loss reduction largely disappears, and losses return close to the IID level.



(a) MNIST



(b) CIFAR-10



(c) Original GR

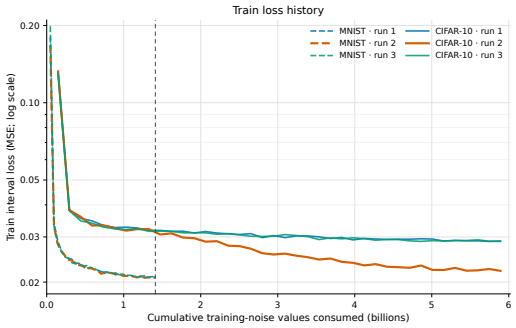


(d) Shuffled GR

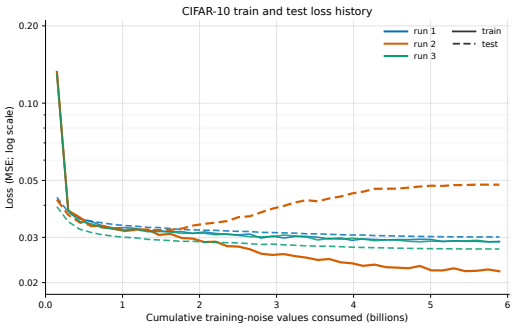


(e) IID

Figure 3: Shuffle ablation on MNIST and CIFAR-10 using run 1. Shuffling preserves the same PRNG values while disrupting their original ordering, and removes most of the source-dependent loss reduction on both datasets. The MNIST generations in (c)–(e) show the same recovery: shuffled GR (d) approaches the IID quality level (e) and is substantially better than original GR (c).



(a) Training loss versus cumulative noise exposure



(b) CIFAR-10 train–test histories

Figure 4: Finite-trajectory effects across runs for SS8 on real-image data. Left: training loss plotted against the cumulative number of training-noise values consumed. CIFAR-10 run 2 separates only after a larger noise exposure and reaches a lower training loss. Right: in CIFAR-10 run 2, the training loss continues to decrease while the same-source test loss increases, producing a clear train–test divergence.

Generation reflects the same change: the displayed MNIST outputs shift from disordered textures to recognizable digits resembling the IID examples. With the values held fixed, these results support the ordering of noise as a major source of the structure learned during training.



(a) GR, IID (b) GR, matched (c) M1, IID (d) M1, matched (e) IID baseline

Figure 5: Matched-source noise during generation. Panels (a,b) use the same GR-trained model, and panels (c,d) use the same M1-trained model, with only the initial noise source changed from IID to the matched source. All four use deterministic DDIM sampling with $\eta = 0$. Panel (e) shows the IID-trained baseline with IID sampling. The improvement under matched-source sampling is consistent with the reuse of source-specific cues learned during training.

4.4 Seed-Dependent Finite-Trajectory Effects

Training consumes a concrete finite noise sequence. Besides regularities shared across trajectories of the same PRNG, this sequence can contain accessible patterns specific to the realized trajectory. When shared regularities no longer dominate the learned noise dependence, exploiting trajectory-specific patterns can further reduce training loss. If these patterns do not hold in the separately seeded test trajectory, reliance on them can instead increase test loss, producing

$$L_{\text{train}}(s) < L_{s \to s}.$$

We examine SS8 on real MNIST and CIFAR-10 images. To compare noise exposure across datasets, we plot losses against cumulative noise consumption, $N_{\text{noise}}(k) = CHW \sum_{j=1}^{k} B_j$, where B_j is the actual batch size and C, H, W specify the image shape. Figure 4 shows the predicted pattern in CIFAR-10 run 2: beyond MNIST's maximum noise exposure, training loss decreases further while same-source test loss rises. Their final tail means are 0.022519 and 0.047631, respectively. The three MNIST runs remain close to one another, as do CIFAR-10 runs 1 and 3.

Under the same run-2 Noise-only protocol, all other 12 sources have decreasing test-loss tail trends and absolute train–test tail gaps at most 0.000176. Shared auxiliary randomness alone therefore does not explain the SS8-specific divergence. Together, these observations support learning that specializes to the realized training trajectory beyond regularities shared across seeds.

4.5 Matched-Source Noise as a Generation Cue

We compare generation from PRNG-trained models using matched-source and IID initial noise. Reliance on learned source-specific regularities creates a denoising mismatch when these regularities are absent from IID inputs. Section 4.2 already demonstrates the prediction cost of this mismatch: $L_{s \to \text{IID}}$ can substantially exceed the IID test reference $L^{(r)}_{0,\text{bank}}$. We therefore predict better generation under matched-source initialization, where the noise supplies information that the model has learned to use as a generation cue.

Figure 5 supports this prediction. Holding each GR- or M1-trained model fixed, we change only the initial noise source and use deterministic DDIM with $\eta = 0$, without additional noise along the reverse trajectory. Matched-source initialization improves the representative MNIST outputs for both models. This benefit concerns compatibility with the trained model rather than an intrinsic superiority of these generators. Modifying the training objective also improves representative outputs under IID initialization (Appendix D). Noise cues can also be deliberately associated with prescribed targets during training to serve as triggers, as exemplified by TrojDiff (Chen et al., 2023), discussed in Section 5.2.

5 Independent Evidence for Noise Learnability and Designability

Controlled interventions in Section 4 establish learned dependence on PRNG noise.

Independent prior results provide complementary evidence beyond this setting: non-OT noise–data assignment weakens a purely distance-based explanation (Li et al., 2024), noise triggers demonstrate learned noise–target associations (Chen et al., 2023), and structured-noise fine-tuning enables motion control in billion-parameter video diffusion models (Burgert et al., 2025). Together, these results show that noise learnability can be deliberately exploited to improve learning and shape generation.

5.1 Noise–Data Assignment: Learnability Beyond Direct Distance Minimization

In Immiscible Diffusion, Li et al. (2024) reassign Gaussian noise tensors to clean images within each training batch and compare the original assignment with a non-optimal-transport (non-OT) control. For a batch of B images x_i and noise tensors ϵ_j, let R reverse the coordinates of a noise tensor. The assignments and resulting training inputs are

$$\pi_A^* = \arg\min_{\pi} \sum_{i=1}^{B} \|x_i - A\epsilon_{\pi(i)}\|_2, \qquad A \in \{I, R\}, \tag{7}$$

$$x_{t,i} = \sqrt{\bar{\alpha}_t}\, x_i + \sqrt{1-\bar{\alpha}_t}\, \epsilon_{\pi_A^*(i)}. \tag{8}$$

Here, π permutes the noise indices, $A = I$ gives the original assignment, and $A = R$ gives the non-OT control. In the latter, coordinate reversal determines the pairing, while the original noise is used in diffusion training (their Sections 3.2 and 4.3).

Both assignments make the selected noise informative about its paired image: one favors direct correspondence, the other correspondence after coordinate reversal. Repeating the assignment throughout training continually presents these relationships to the model through its noisy input. Both methods accelerate training and improve generation quality relative to the baseline at comparable training steps, with nearly comparable FID values in the later stages. Thus, the benefit persists when direct distance minimization is removed but image-dependent information remains available through a transformed correspondence. This provides evidence that the model learns to use information carried by noise–data pairing to support prediction. The reversed correspondence may take longer to become useful because the model must learn to exploit a different arrangement of the relationship. Once learned, this correspondence can likewise provide useful image-related information. This explanation is consistent with the non-OT ablation (their Section 4.3, Figure 7), in which the OT and non-OT variants achieve nearly comparable FID values in the later stages of training.

5.2 Noise Triggers: Learned Target Associations

TrojDiff (Chen et al., 2023) introduces triggers into the noise inputs used during training and sampling. For its blend-based trigger, the noisy training input is constructed as

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}(\mu + \gamma\epsilon), \tag{9}$$

where $\mu = (1-\gamma)\delta$, δ is the trigger, and $\epsilon \sim \mathcal{N}(0, I)$. The method combines Trojan training on samples from a chosen target distribution with standard training on the original data. During generation, triggered sampling starts from $x_T \sim \mathcal{N}(\mu, \gamma^2 I)$ and follows the corresponding Trojan reverse process (their Sections 3.1–3.3).

During Trojan training, the trigger carried by noise is paired with a prescribed target: an in-domain class, an out-of-domain class, or a single specified image. During generation, models trained with these pairings can produce the corresponding targets from triggered noise under the Trojan sampling procedure (their Section 4.2, Tables 1 and 2). The realization of the association specified during training provides evidence that the model learns to use information introduced through noise. Through this learned association, the trigger serves as guidance toward the target during generation. Their training-step ablation further supports acquisition of this association: under blend-based In-D2D attacks on CIFAR-10 evaluated with DDIM, attack success rises from 0% at 20k fine-tuning steps to 85.9% at 50k, while benign-generation performance remains stable (their Section 4.3, Figure 3).

5.3 Structured Video Noise: Learnability at Billion-Parameter Scale

Go-with-the-Flow places motion information in the noise used to fine-tune a video diffusion model (Burgert et al., 2025). During fine-tuning, each training video's optical flow is used to warp Gaus-

sian noise across frames, creating temporal correspondences while preserving spatial Gaussianity within each frame. This noise is added to the clean video to form the denoising input, pairing noise structure with the video's motion while retaining the original architecture and training objective (their Section 3.3). During generation, desired motion is encoded in warped noise that initializes the reverse process of the already fine-tuned model (their Section 3.4).

Their comparison distinguishes using warped noise only during generation from using it during both fine-tuning and generation. With generation-only use, the original CogVideoX model roughly follows the motion encoded in the noise, but exhibits visual artifacts and degraded frame quality. With warped noise used in both stages, the fine-tuned model produces coherent motion-controlled outputs in the reported comparison (their supplementary Section 9, Figure 14). Since both settings use warped noise during generation, this comparison supports the contribution of training to its effective use. Our interpretation is that fine-tuning teaches the model to associate temporal relationships in the noise with motion in clean videos; during generation, these learned associations allow the initial noise to serve as motion guidance. The degradation experiment further examines this generation-time role: for the fine-tuned model, retaining more warped structure in the initial noise makes the generated videos follow the supplied flow more closely (their Section 3.3, Figure 4).

A unified interpretation. Although these methods differ in their designs and objectives, their results provide complementary evidence for a common learning principle: models can learn to exploit structure entering through noise and its relationships with clean data. These designs demonstrate that noise learnability can be deliberately used to shape model behavior, including improving learning efficiency and generation control. Noise is therefore a designable input dimension through which useful information can be introduced to enhance model capabilities.

6 Beyond Diffusion Models

The learning view in Section 2.1 extends beyond diffusion models: concrete random inputs enter the realized loss and its gradient, and training can learn to exploit patterns in these inputs when doing so helps reduce the loss. Existing observations provide suggestive examples. Naruse et al. (2019) reported that properties associated with the temporal structure of chaotic GAN inputs were reflected in the generated images; the corresponding signature disappeared after the input sequence was shuffled. Koivu et al. (2022) observed changes in fitted models when different random-number generators were used to control dropout.

A concrete question is whether large generative models already learn to exploit patterns in widely used PRNG outputs, and whether this affects generation. The capacity to learn from random inputs also creates an opportunity for deliberate design: useful structure can be introduced or preserved so that learning from randomness contributes to better model performance.

7 Conclusion

Diffusion models can learn to exploit structure in noise, making noise a learnable and designable input that affects learning and generation. This learnability can be deliberately used to introduce useful structure through noise and improve model performance.

8 Limitations

Our controlled experiments are limited to relatively small diffusion models on MNIST and CIFAR-10; evidence in larger diffusion models comes from the prior studies discussed in Section 5. We have not directly tested whether existing large generative models learn structure in industrial-grade PRNG outputs in ways that affect generation quality. The general learning view in Section 2.1 has not been systematically tested in learning systems other than diffusion models. Systematic principles for selecting and designing useful noise structure across tasks and architectures also remain to be developed.

Reproducibility Statement

Section 3 and Appendix A describe the noise sources, preprocessing, random-role routing, seed protocol, and training and sampling settings. Appendix C derives the IID references, and Appendix D specifies the modified objective. Experiments were run with Python 3.12.3 and PyTorch 2.8.0 (CUDA 12.8) on NVIDIA RTX 4090 and RTX 5090 GPUs. Noise generation used NumPy 2.3.2 and Numba 0.67.0 with the Numba backend enabled. The anonymous supplementary package provides the experiment code, configurations, recorded results, and analysis scripts, with environment and reproduction instructions in its README.

AI Use Statement

The authors used OpenAI ChatGPT and Codex for literature search; conceptual consistency checking and assistance with mathematical formulation; feedback on methodology and experimental design; interpretation of results; code implementation and debugging; numerical cross-checking; English translation; and drafting, revising, organizing, and polishing portions of the manuscript.

The core research idea originated with the authors, who directed the study and made the final decisions on the methodology, scientific claims, and conclusions. The authors reviewed and checked AI-assisted text, references, code, and analyses and take full responsibility for the final content and results of this work.

References

Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, Michael Ryoo, Paul Debevec, and Ning Yu. Go-with-the-Flow: Motion-controllable video diffusion models using real-time warped noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13–23, 2025.

Weixin Chen, Dawn Song, and Bo Li. TrojDiff: Trojan attacks on diffusion models with diverse targets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4035–4044, 2023.

Giannis Daras, Yuval Dagan, Alexandros G. Dimakis, and Constantinos Daskalakis. Consistent diffusion models: Mitigating sampling drift by learning to be consistent. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, PA, second edition, 2002.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Xingchang Huang, Corentin Salaün, Cristina Vasconcelos, Christian Theobalt, Cengiz Öztireli, and Gurprit Singh. Blue noise for diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024. Article 28.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

M. Klöwer, P. V. Coveney, E. A. Paxton, and T. N. Palmer. Periodic orbits in chaotic systems simulated at low precision. *Scientific Reports*, 13:11410, 2023.

Aki Koivu, Joona-Pekko Kakko, Santeri Mäntyniemi, and Mikko Sairanen. Quality of randomness and node dropout regularization for fitting neural networks. *Expert Systems with Applications*, 207:117938, 2022.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Shujun Li, Guanrong Chen, and Xuanqin Mou. On the dynamical degradation of digital piecewise linear chaotic maps. *International Journal of Bifurcation and Chaos*, 15(10):3119–3151, 2005.

Yiheng Li, Heyang Jiang, Akio Kodaira, Masayoshi Tomizuka, Kurt Keutzer, and Chenfeng Xu. Immiscible diffusion: Accelerating diffusion training with noise assignment. In *Advances in Neural Information Processing Systems*, volume 37, pp. 90198–90225, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Eliya Nachmani, Robin San Roman, and Lior Wolf. Non-Gaussian denoising diffusion models. *arXiv preprint arXiv:2106.07582*, 2021.

Makoto Naruse, Takashi Matsubara, Nicolas Chauvet, Kazutaka Kanno, Tianyu Yang, and Atsushi Uchida. Generative adversarial network based on chaotic time series. *Scientific Reports*, 9:12963, 2019.

Edward Ott. *Chaos in Dynamical Systems*. Cambridge University Press, Cambridge, UK, second edition, 2002.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

Tao Tao, Darshil Doshi, Dayal Singh Kalra, Tianyu He, and Maissam Barkeshli. (How) can transformers predict pseudo-random numbers? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59103–59131, 2025.

Vikram Voleti, Christopher Pal, and Adam Oberman. Score-based denoising diffusion with non-isotropic Gaussian noise models. In *NeurIPS 2022 Workshop on Score-Based Methods*, 2022.

A Experimental Protocol Details

A.1 Random sources and preprocessing

The main experiments use the following source families. For the Logistic family,

$$x_{n+1} = a x_n (1 - x_n), \qquad a \in \{3.7, 3.8, 3.9\}.$$

For the higher-order Markov-type sources,

$$x_{n+1} = \left(\sum_{j=1}^{k} \frac{j}{k} x_{n-k+j} + 0.123456 \right) \bmod 1, \qquad k \in \{2, 4, 8\},$$

while the first-order source is

$$x_{n+1} = (3.912345 x_n + 0.17321) \bmod 1.$$

The ShiftedSine family is

$$x_{n+1} = (a \sin(\pi x_n) + c) \bmod 1,$$

with

$$a \in \{1, 2, 4, 8\}, \qquad c = 0.36787944117144233.$$

We additionally use a reimplementation of the degree-31, separation-3 GLIBC `random()` generator, denoted GR. NumPy's `default_rng` is used as the operational IID reference.

Each deterministic source undergoes 1,000 protocol-level burn-in iterations before downstream use. To construct the fixed marginal transform, a separately seeded reference trajectory of length

$$N_{\mathrm{ref}} = 100000$$

is generated and sorted. For a raw value x, let $l(x)$ and $r(x)$ denote its left and right insertion indices in the sorted reference trajectory. We define the empirical mid-rank value

$$u(x) = \frac{\frac{1}{2}\big(l(x) + r(x)\big) + \frac{1}{2}}{N_{\mathrm{ref}} + 1}.$$

Before the inverse-normal transform, u is clipped to $[\delta, 1 - \delta]$, where $\delta = 0.5/(N_{\mathrm{ref}} + 1)$. The Gaussianized value is then

$$z(x) = \Phi^{-1}(u(x)).$$

A second, separately seeded trajectory of length

$$N_{\mathrm{cal}} = 200000$$

is transformed in the same way to estimate its mean μ and standard deviation σ. The final value used by the diffusion-noise stream is

$$\widetilde{z}(x) = \frac{z(x) - \mu}{\sigma + 10^{-12}}.$$

The empirical-rank reference trajectory, affine-calibration trajectory, and downstream experimental trajectories are generated using separate seeds. The resulting map is fixed before downstream training and is applied pointwise; ranks are not recomputed within individual batches or tensors. Therefore the preprocessing modifies the one-dimensional marginal without reordering the source trajectory.

The IID source bypasses empirical ranking: its uniform draws undergo the same clipping, inverse-normal transform, and independently seeded affine calibration. Calibration uses the population standard deviation. Floating-point source recurrences and marginal transforms use float64; downstream streams are stored as float32. Uniform streams used for auxiliary roles and IID clean tensors apply the same clipping but not the inverse-normal or affine transform.

A.2 RANDOM-ROLE ROUTING, PROBE CONSTRUCTION, AND OPTIMIZATION

Each diffusion run is defined by one master seed. The three master seeds used for the reported runs are

$$1827654268, \qquad 119067221, \qquad 235869037.$$

Role-specific child seeds are deterministically derived for individual random roles. In the main diffusion protocol, the child seed assigned to a role depends on the run and the role identifier but not on the tested source name. Thus, replacing one tested source by another changes the generator rule while keeping the corresponding role seed matched.

The supplied code specifies seed derivation in `seeding.py`: `SeedContext` hashes the protocol version, master-seed salt, and role with SHA-256, and uses the first eight digest bytes in little-endian order. Integer source seeds are reduced modulo $2^{31} - 2$ and incremented by one; scalar initial states are mapped to $[0.05, 0.95)$. The corresponding initialization rules, including higher-order states and GR warm-up, are specified in `sources.py`.

The random roles used in the ablation are routed as follows:

Profile	Train noise	Test noise	Auxiliary roles	Sampling
Full PRNG	Tested	Tested	Tested	IID
Noise only	Tested	Tested	IID	IID
Auxiliary only	IID	IID	Tested	IID

The auxiliary roles include model initialization, training and evaluation timestep sampling, and training-data shuffling. Sampling is deliberately outside these three routing profiles. Under the standard protocol, initial sampling noise is IID and deterministic DDIM with $\eta = 0$ consumes no stepwise sampling noise.

Training and evaluation diffusion noise use different child seeds to generate separately seeded trajectories from the same source family.

For the IID-clean diffusion probe, clean data has its own dedicated child seed, with separate train and evaluation streams derived beneath it. Scalar clean values are generated from the operational IID uniform stream,

$$u \in [0, 1],$$

and mapped to

$$x_0 = 2u - 1.$$

The resulting tensors have the same shape and value range as the corresponding real-data inputs. Probe training clean tensors are generated online from a continuous stream. Probe evaluation uses a separately seeded fixed bank of 4,096 clean tensors. The diffusion noise is generated independently from the tested source and enters through the standard forward process

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t.$$

The model is trained with the same epsilon-prediction objective as in the real-data experiment.

Real images are scaled from integer pixels to $[-1, 1]$ by division by 127.5 and subtraction of one, without augmentation. Each real-data evaluation bank selects 4,096 test images without replacement using its dedicated IID child seed. Auxiliary timestep draws are mapped to $\lfloor Tu \rfloor$ and clipped to $\{0, \ldots, T - 1\}$. Model initialization uses seeded PyTorch defaults for IID and the routed uniform stream otherwise with the usual fan-in bounds for linear and convolutional layers; normalization parameters retain their deterministic defaults.

The MNIST U-Net uses channel widths

$$(64, 128, 128),$$

with input shape

$$1 \times 28 \times 28.$$

The CIFAR-10 U-Net uses channel widths

$$(128, 256, 256, 256),$$

with input shape

$$3 \times 32 \times 32.$$

All models are trained from scratch. MNIST uses all 60,000 training images with batch size 128 for 30 epochs (14,070 optimization steps). CIFAR-10 uses all 50,000 training images with batch size 64 for 30,000 steps. Final batches of a data pass are retained even when smaller than the nominal batch size. These settings apply to the corresponding real-data and IID-clean comparisons. For the source-matched and source-to-IID comparisons, training was executed separately with matched configurations and seeds. Deterministic GPU execution was not enforced, so matching the seeds did not guarantee identical training trajectories or losses. Each reported training and test loss comes from its corresponding execution.

Both datasets use a linear diffusion schedule with

$$\beta_1 = 10^{-4}, \qquad \beta_T = 0.02, \qquad T = 1000.$$

The implementation uses `diffusers==0.39.0` and `UNet2DModel`, with `layers_per_block=2` and no dropout. Both U-Nets use attention in the second downsampling block and the corresponding upsampling block, as well as the middle block. MNIST uses eight normalization groups; CIFAR-10 uses 32 groups, normalization epsilon 10^{-6}, zero downsampling padding, a frequency shift of one, and `flip_sin_to_cos=False`. All constructor arguments are retained in `diffusion_models.py`.

Optimization uses AdamW (Loshchilov & Hutter, 2019) with learning rate

$$10^{-4},$$

momentum parameters

$$(0.95, 0.999),$$

weight decay

$$10^{-6},$$

a 500-step warm-up, cosine learning-rate decay, and gradient-norm clipping at 1.0. Mixed-precision training is used when CUDA is available.

MNIST is evaluated every 469 optimization steps and CIFAR-10 every 750 steps. Each evaluation reuses the same fixed clean examples, timesteps, and noise tensors for all checkpoints within a run. Test MSE is averaged over all evaluated noise elements, and test tail loss is the arithmetic mean of the final 10 recorded evaluation losses. The ideal probe reference uses the same run-specific timestep weights (Appendix C); these weights are not replaced by a uniform average when comparing test losses.

Each recorded training loss is the example-weighted average of minibatch losses over the preceding evaluation interval, evaluated before each parameter update. Training tail loss averages the final ten such records; it is not a loss reevaluated at one fixed final checkpoint.

Standard generation produces 1,000 samples using 100-step DDIM sampling with

$$\eta = 0$$

and IID initial noise. Experiments that replace the initial sampling source by the matched tested source are treated separately from the standard training protocol.

A.3 Shuffle-ablation implementation

The shuffle ablation is applied after the source-specific marginal transform. For each tested diffusion-noise role, the complete scalar stream that would otherwise be consumed by that role is first materialized as a one-dimensional tape,

$$(z_1, z_2, \ldots, z_N).$$

An independent global permutation π is then applied,

$$(z_1, z_2, \ldots, z_N) \longrightarrow (z_{\pi(1)}, z_{\pi(2)}, \ldots, z_{\pi(N)}).$$

The shuffled and original streams therefore contain exactly the same scalar values with exactly the same empirical marginal distribution. Only their positions and sequential dependence are changed.

The permuted tape is subsequently consumed in the same order as an ordinary stream and reshaped into diffusion-noise tensors as required by the training or evaluation procedure.

Permutation is role-local. Training-noise and evaluation-noise tapes are shuffled independently, and values are never exchanged between distinct random roles. The permutation itself is generated by an independent PCG64DXSM stream with a role-specific child seed and does not use the tested source to determine the ordering.

For large tapes, the implementation uses a bounded-memory external permutation procedure, but this affects only memory management and not the experimental definition above.

B Source Screening: Absorbing States and Period Reuse

B.1 Finite-precision absorbing states

We exclude trajectories that collapse into absorbing states under finite-precision arithmetic. The numerical iteration is a map on machine-representable values rather than the exact real-valued dynamical system, and rounding can therefore create trajectories that do not occur in exact arithmetic (Ott, 2002; Higham, 2002; Li et al., 2005; Klöwer et al., 2023).

The risk is particularly large near a critical point c satisfying

$$f'(c) = 0.$$

For a small displacement h,

$$f(c+h) = f(c) + \frac{1}{2}f''(c)h^2 + O(h^3).$$

If Δ denotes the effective output-rounding tolerance, values satisfying

$$|f(c+h) - f(c)| \lesssim \Delta$$

can be mapped to the same machine-representable output. Neglecting higher-order terms gives the approximate capture width

$$|h| \lesssim \sqrt{\frac{2\Delta}{|f''(c)|}}.$$

By contrast, near a point with

$$f'(c) \neq 0,$$

the corresponding width is approximately

$$|h| \lesssim \frac{\Delta}{|f'(c)|}.$$

A critical point can therefore enlarge the rounding-induced capture region from order Δ to order $\sqrt{\Delta}$.

For the Sine-1 map,

$$f(x) = \sin(\pi x) \bmod 1,$$

the experiment reached

$$x = 0.5000000021116002.$$

The floating-point evaluation of

$$\sin(\pi x)$$

rounded to exactly 1. The modulo operation then produced 0, after which the orbit remained permanently at the zero absorbing state.

For Logistic 4.0,

$$f(x) = 4x(1-x),$$

the trajectory reached

$$x = 0.4999999944571299.$$

The next floating-point evaluation rounded to exactly 1.0, after which the following iteration produced 0.0 and the orbit remained at zero.

These failures arise from the realized finite-precision trajectories rather than from the continuous comparison of source learnability studied in the main experiments. Sine-1 and Logistic 4.0 are therefore excluded from the main source set.

B.2 Period reuse under the training budget

PRBS31 is excluded for a different reason. Its nonzero 31-bit state has at most $2^{31} - 1 = 2{,}147{,}483{,}647$ possible values, and the implementation advances one state per output scalar. The CIFAR-10 training-noise stream consumes $5{,}892{,}636{,}672$ values, exceeding that state count and necessarily reusing states. This is period reuse, not entry into an absorbing state or an inability to generate further values. MNIST consumes $1{,}411{,}200{,}000$ training-noise values. Counts use actual batch sizes, $N_{\text{noise}}(k) = CHW \sum_{j=1}^{k} B_j$, and do not combine separate training, evaluation, sampling, or calibration streams. PRBS31 is omitted consistently from the displayed comparisons and processed summary tables; original records are retained.

C Derivation of the Ideal IID Probe References

At timestep t, one coordinate of the ideal probe is $x = a_t u + b_t e$, with independent $u \sim \text{Uniform}(-1, 1)$ and $e \sim \mathcal{N}(0, 1)$, where $a_t = \sqrt{\bar{\alpha}_t}$ and $b_t = \sqrt{1 - \bar{\alpha}_t}$. Coordinate independence reduces the elementwise tensor MSE to this scalar problem. Given x, e is standard normal truncated to $[l_t(x), h_t(x)]$, where $l_t(x) = (x - a_t)/b_t$ and $h_t(x) = (x + a_t)/b_t$. Writing ϕ and Φ for the standard normal density and CDF,

$$\begin{aligned} m_t(x) = \mathbb{E}[e \mid x, t] &= \frac{\phi(l_t(x)) - \phi(h_t(x))}{\Phi(h_t(x)) - \Phi(l_t(x))}, \\ p_t(x) &= \frac{\Phi(h_t(x)) - \Phi(l_t(x))}{2a_t}. \end{aligned} \tag{10}$$

Since the conditional mean minimizes squared loss and $\mathbb{E}[e^2] = 1$, the per-timestep risk is

$$L_t^* = \mathbb{E}[(e - m_t(x))^2] = 1 - \int_{-\infty}^{\infty} m_t(x)^2 p_t(x)\, dx. \tag{11}$$

We evaluate this one-dimensional integral numerically for the 1,000-step linear schedule, with β_t increasing from 10^{-4} to 0.02 and $\bar{\alpha}_t = \prod_{j=0}^{t}(1 - \beta_j)$. The uniform average in Equation 6 gives $L_0 = 0.1840158408\ldots$.

Evaluation instead averages these same risks at the run's 4,096 fixed timesteps, counting repeated timesteps each time they occur. This gives $L_{0,\text{bank}}^{(r)}$ in Equation 6. The bank is shared across noise sources and image shapes within a run, yielding the references in Table 1.

Table 1: Run-specific ideal IID references and measured IID test losses. Each reference averages the per-timestep risks using that run's 4,096 evaluation timesteps, while retaining ideal IID clean and Gaussian-noise distributions within each timestep.

Run	Ideal IID, test timesteps	MNIST IID test tail	CIFAR-10 IID test tail
1	0.187006	0.187440	0.187342
2	0.183548	0.183833	0.183816
3	0.174592	0.174871	0.174701

The run-specific references account for most of the differences between the measured IID test losses; a run-3 loss below 0.184016 uses different timestep weights, rather than exceeding its own ideal reference. These calculations average over ideal clean/noise distributions and are reference risks, not hard bounds on finite empirical losses. The accompanying code includes `analysis/calc_l0_reference.py`, the seed reconstruction, numerical integration settings, and exported timestep risks and weights.

D Modified-Objective Ablation and CDM Loss Adaptation

This supporting ablation asks whether changing the objective alters generation under IID sampling. It retains noise-prediction MSE and jointly adds clean-prediction consistency and a two-sample

clean-target term. Figure 6 shows improved representative GR and M1 outputs. Because both additions are used together, the comparison does not identify their separate effects or measure a division between clean-data learning and noise-structure use. The objective and implementation below specify this sensitivity check.



(a) GR, standard (b) GR, modified (c) M1, standard (d) M1, modified (e) IID baseline

Figure 6: Modified-objective ablation with IID initial noise throughout. Panels (a,b) compare standard and modified objectives for GR; (c,d) give the corresponding M1 comparison. The modified objective is defined in Equation 15. Panel (e) shows the standard-objective IID-trained baseline. These representative MNIST examples illustrate the dependence of generation behavior on the training objective.

The modified-loss experiment uses a discrete VP adaptation of the consistency regularization in Consistent Diffusion Models (Daras et al., 2023). For a noisy input x_t, the model first predicts $\epsilon_\theta(x_t, t)$ and obtains the corresponding unclipped clean estimate

$$\widetilde{h}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}, \qquad h_0 = \mathrm{clip}(\widetilde{h}_0, -1, 1). \tag{12}$$

From the same x_t, two independent stochastic reverse trajectories are then generated to $s = \max(t - \Delta, 0)$, with Δ drawn uniformly from $\{1, \ldots, 50\}$; thus $s < t$ unless $t = 0$. Let $\widetilde{h}_1$ and $\widetilde{h}_2$ denote the unclipped clean estimates obtained at the two trajectory endpoints, and define $h_r = \mathrm{clip}(\widetilde{h}_r, -1, 1)$ for $r \in \{1, 2\}$. The consistency term is

$$\mathcal{L}_{\mathrm{CDM}} = \frac{1}{d}\sum_{j=1}^{d}(h_{1,j} - h_{0,j})(h_{2,j} - h_{0,j}). \tag{13}$$

The two trajectories are reused for the additional clean-image term. Using the unclipped endpoint clean estimates $\widetilde{h}_1$ and $\widetilde{h}_2$, we define

$$\mathcal{L}_{x_0} = \frac{1}{2d}\sum_{j=1}^{d}(\widetilde{h}_{1,j} - x_{0,j})(\widetilde{h}_{2,j} - x_{0,j}). \tag{14}$$

Because the two endpoint trajectories are conditionally independent given (x_t, t, s, x_0), this cross-product is a two-sample estimator of the squared deviation of the endpoint conditional mean from the clean target x_0, up to the factor $1/2$. Unlike an ordinary endpoint MSE, it does not directly penalize the conditional variance of the endpoint predictions. The complete training objective is

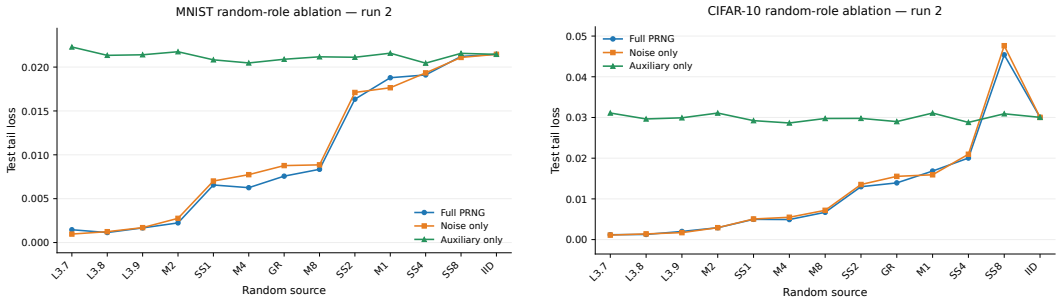
$$\mathcal{L} = \mathcal{L}_\epsilon + 2\mathcal{L}_{\mathrm{CDM}} + 0.1\mathcal{L}_{x_0}, \tag{15}$$

where $\mathcal{L}_\epsilon$ is the standard noise-prediction MSE.

In the experiment, the local reverse rollout uses six discrete points, a maximum timestep drop of 50, and stochastic DDIM transitions with $\eta = 1$. The original CDM consistency term uses clipped clean estimates in the range $[-1, 1]$, while the additional x_0 term uses the corresponding unclipped endpoint estimates. The six points are obtained by rounding a linearly spaced grid from t to s, giving five transitions; repeated timestep indices are allowed. Rollout noise comes from an independent IID PyTorch generator with the role seed `loss/cdm_rollout`. Rollout states are detached and generated without gradient tracking; gradients are retained through the model evaluations used for h_0 and the endpoint estimates. The displayed modified-objective comparison uses MNIST run 1 with the same training budget and IID generation protocol as the standard objective.

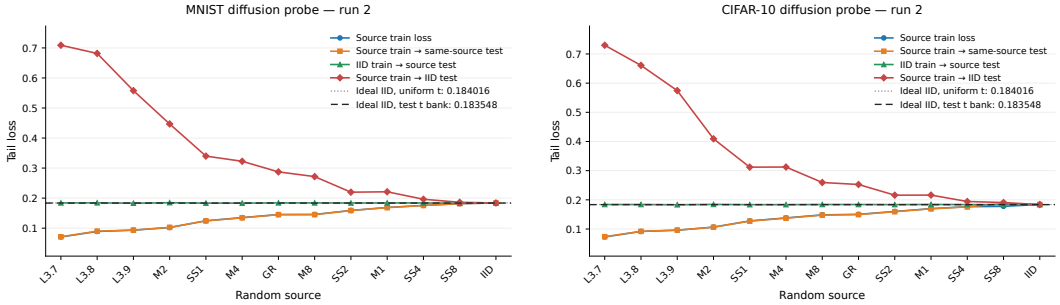
E ADDITIONAL RUNS: ROLE ABLATION AND IID-CLEAN PROBE

The following figures report runs 2 and 3 separately for the experiments in Sections 4.1 and 4.2. Each comparison retains 13 sources after excluding PRBS31. The probe figures include the corresponding run-specific evaluation references derived in Appendix C.



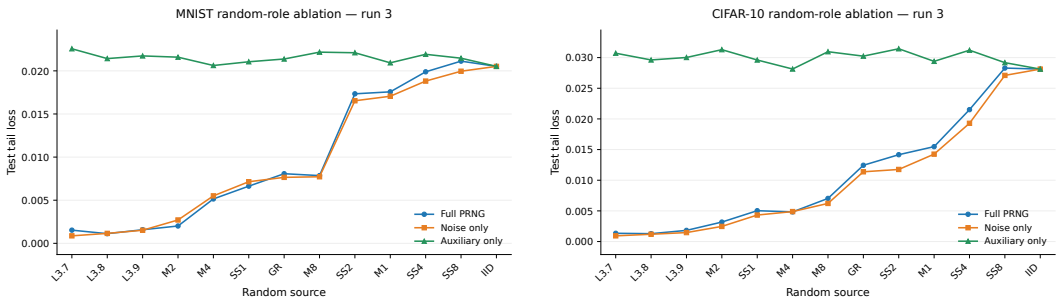
(a) MNIST (b) CIFAR-10

Figure 7: Random-role ablation, run 2. Test tail loss for Full PRNG, Noise-only, and Auxiliary-only; sources are ordered by Noise-only loss, with IID placed last. Each curve uses the same run and dataset.



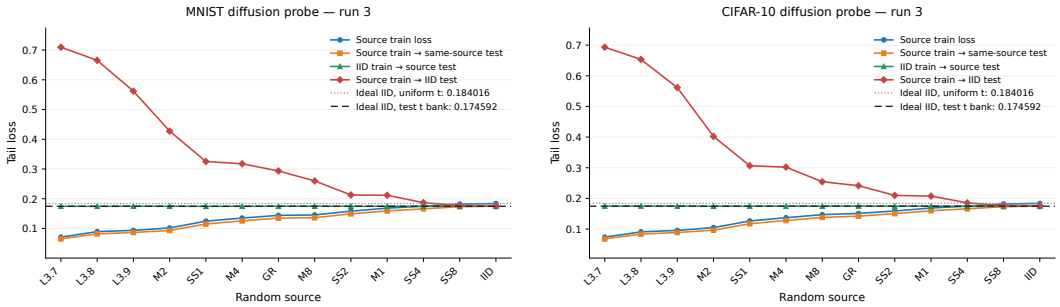
(a) MNIST (b) CIFAR-10

Figure 8: IID-clean diffusion probe, run 2. The four curves show source training loss, matched-source test loss, IID-trained/source-tested loss, and source-trained/IID-tested loss. The dotted line is the uniform-timestep ideal IID reference (0.184016); the dashed line uses the evaluation timesteps for this run (0.183548). Sources are ordered by training loss.



(a) MNIST (b) CIFAR-10

Figure 9: Random-role ablation, run 3. Test tail loss for Full PRNG, Noise-only, and Auxiliary-only; sources are ordered by Noise-only loss, with IID placed last. Each curve uses the same run and dataset.



(a) MNIST (b) CIFAR-10

Figure 10: IID-clean diffusion probe, run 3. The four curves show source training loss, matched-source test loss, IID-trained/source-tested loss, and source-trained/IID-tested loss. The dotted line is the uniform-timestep ideal IID reference (0.184016); the dashed line uses the evaluation timesteps for this run (0.174592). Sources are ordered by training loss.