EQUIVALENT FLOWS, UNEQUAL LEARNING: CLEAN-LATENT PREDICTION IN TRANSFORMERS

Anonymous authors
Paper under double-blind review

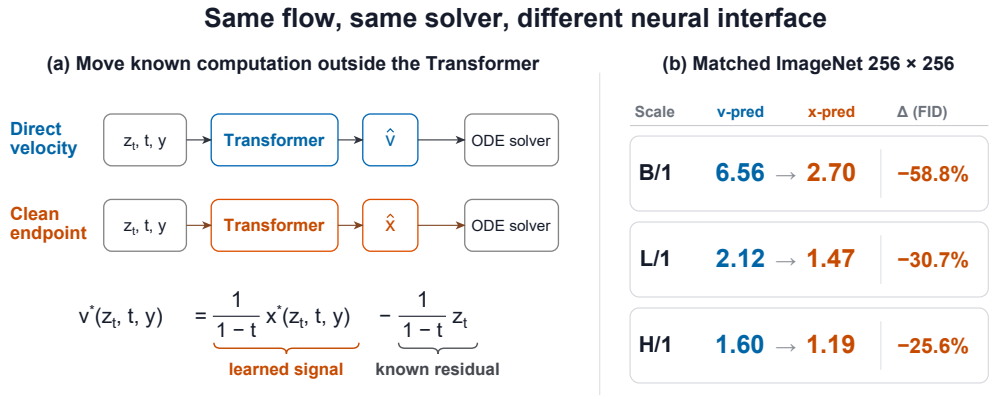

Figure 1: **The solver interface need not be the neural interface.** (a) Direct velocity prediction learns the vector field internally, whereas clean-endpoint prediction exposes $\hat{x}$ directly to the Transformer and shifts the linear residual response $\hat{v} = (\hat{x} - z_t)/(1-t)$ to an exact readout, leaving the ODE solver interface identical. (b) ImageNet 256×256 FID across B/1, L/1, and H/1 scales (Table 1).

ABSTRACT

Flow samplers consume velocity, but the neural network can predict the clean endpoint and convert it to velocity through a fixed affine readout. We study this choice with JLT, a latent Transformer in a frozen variational autoencoder (VAE) representation. For squared error, the optimal clean and velocity predictors are algebraically equivalent; a finite Transformer assigns different computation to its learned output under the two interfaces. A local Gaussian analysis identifies a known residual response supplied by the readout and isotropic target variance added by velocity prediction. Measured FLUX.2 channel spectra support this geometric distinction: 90% of target variance occupies 83 of 128 clean directions versus 109 velocity directions. Under a matched velocity objective, clean prediction improves ImageNet FID-50K from 6.56 to 2.70 at Base scale and from 2.12 to 1.47 at Large scale, with lower FID at every measured Large checkpoint. Scaling clean prediction to 951M parameters reaches FID-50K 1.19 and IS 271.96. In addition, an objective ablation at Base scale shows that direct clean regression reaches FID-50K 2.38 without time-dependent error weighting. These results show how moving known computation outside the network changes learning under algebraically equivalent flow interfaces. Code: https://anonymous.4open.science/r/JLT-0482/README.md

1 INTRODUCTION

An ordinary differential equation (ODE) solver needs a vector field. A neural network does not necessarily need to predict one. Modern flow models often make these two interfaces identical: the solver consumes velocity, and the network is trained to predict velocity. For the linear paths $z_t = tx + (1-t)\epsilon$, however, a clean-endpoint prediction determines exactly the same velocity,

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.

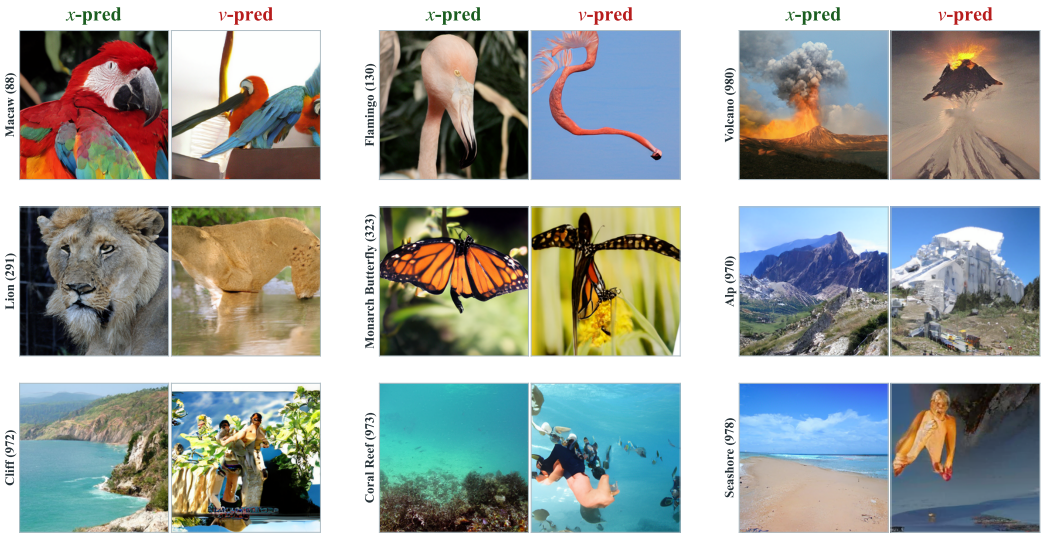

Figure 2: **Qualitative comparison of clean-x and velocity-v prediction under matched Base scale.** Randomly sampled class-conditional generations from matched B/1 models on ImageNet 256×256 with 50-step Heun sampling. The paired samples illustrate differences in global structure and sharpness.

$\hat{v}_x = \frac{\hat{x}-z_t}{1-t}$. At the level of the ideal flow, the two parameterizations are therefore equivalent. Our question is different: **does this equivalence survive the replacement of an ideal predictor by a finite neural network?** For finite networks, algebraically equivalent outputs can induce different learned function classes. The affine conversion contains a known input residual and a time-dependent gain. A network that predicts v must represent this response internally, whereas a network that predicts x can delegate it to the analytic readout. More generally, if a fixed transformation maps one valid output to another, applying that transformation outside a finite learned function need not produce the same parameterized family as learning the transformed quantity directly. Given a corrupted state z_t and time t, a prediction of the clean endpoint x can be converted exactly into velocity by a fixed affine map. The generative flow can therefore remain unchanged while the quantity directly approximated by the neural network changes.

This distinction matters because algebraic equivalence is a statement about outputs, not about the finite parameterized functions used to produce them. For squared error, let $x^*(z_t, t, y)$ denote the optimal clean predictor. The corresponding optimal velocity predictor satisfies

$$v^*(z_t, t, y) = \frac{x^*(z_t, t, y) - z_t}{1 - t}. \tag{1}$$

A direct-velocity Transformer approximates the entire right-hand side as one learned function. A clean-output Transformer can instead approximate x^* and delegate the known input-dependent residual and time-dependent gain to the analytic readout. The solver receives the same quantity either way; what changes is the boundary between learned and known computation. Figure 1 summarizes this view.

Target geometry gives a second reason for the two neural interfaces to behave differently. Clean data in a learned representation can be strongly anisotropic. Velocity $v = x - \epsilon$ combines that structure with unit Gaussian noise, adding variation in every coordinate direction. In a local Gaussian model, the optimal clean predictor suppresses directions in which the clean latent has little variance, whereas the optimal velocity predictor retains a large negative residual response there. The analytic readout supplies this response exactly, leaving the Transformer to model the structured endpoint.

Clean prediction has recently proved effective in pixel-space denoising Transformers (Li & He, 2026), and related work studies generalized prediction targets (Jin & Wang, 2026; Gagneux et al., 2026) and clean prediction in semantic representation spaces (Zhang et al., 2026). Our focus is the latent-flow setting in which generation takes place in a reconstruction-trained VAE representation (Rombach et al., 2022). This setting is useful for isolating the parameterization question: the representation, Transformer architecture, and flow objective can be held fixed while the neural output interface changes.

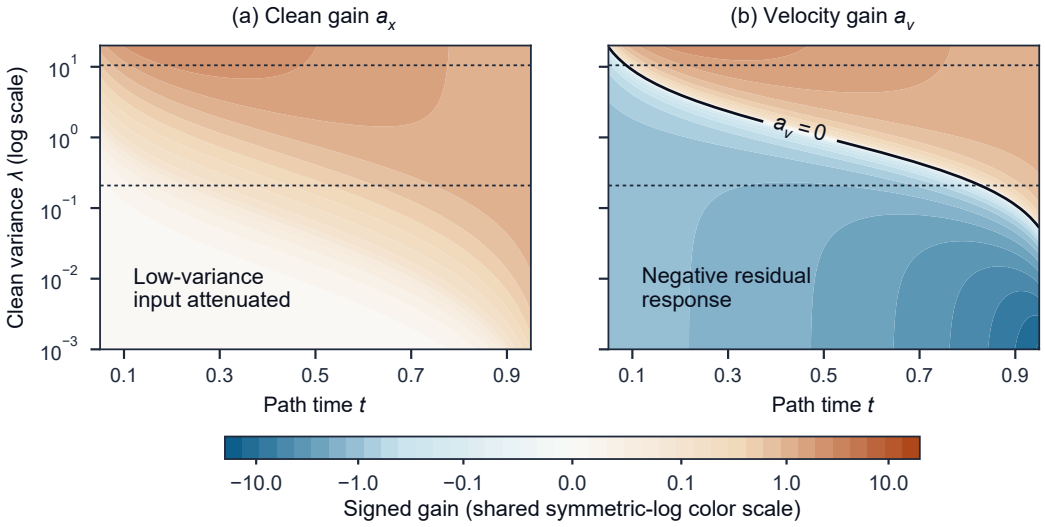

Figure 3: **The analytic readout supplies the residual response in low-variance directions.** Conditional-mean gains $a_x = t\lambda/[t^2\lambda + (1-t)^2]$ and $a_v = [t\lambda - (1-t)]/[t^2\lambda + (1-t)^2]$ for clean and velocity prediction, respectively, over path time $0 < t < 1$ and clean variance λ (derived in Proposition 1). Both panels share a signed symmetric-log color scale. The solid curve marks zero velocity gain; horizontal dotted lines mark the measured FLUX channel-variance range, 0.21 to 10.50. As clean variance decreases, the clean predictor attenuates the input while the velocity response becomes increasingly dominated by the known residual term.

We instantiate this idea with JLT, a latent Transformer that predicts the clean latent and converts it analytically to velocity.[1] On ImageNet 256×256 with a frozen FLUX.2 VAE, matched Base models improve FID-50K from 6.56 with direct velocity to 2.70 with clean output, and matched Large models improve from 2.12 to 1.47. Figure 2 shows paired random samples from the matched B/1 models. At Huge scale (951.3M parameters), clean prediction achieves FID-50K 1.19 and IS 271.96. Across scales, the measured FLUX latent spectrum matches the geometric premise of the analysis: clean variation is concentrated, while adding the velocity noise floor spreads variance across substantially more channel directions.
Our contributions are threefold:

- We separate the neural prediction interface from the flow solver interface. For linear flows, an exact clean-to-velocity readout moves known residual computation outside the Transformer while preserving the same vector field consumed by the sampler.
- We characterize how this reparameterization changes the learned function and target geometry. A local Gaussian analysis exposes the low-variance residual response, and measurements on frozen FLUX.2 latents show the predicted covariance shift and loss of spectral concentration.
- Under matched velocity objectives, clean prediction improves FID at Base and Large scales. At Huge scale, the clean-output model reaches FID-50K 1.19 and IS 271.96.

2 Related Work

Prediction targets and objectives. Diffusion models have long admitted multiple prediction parameterizations. Denoising diffusion probabilistic models popularized noise prediction (Ho et al., 2020; Nichol & Dhariwal, 2021), while velocity parameterization was introduced in the context of progressive distillation (Salimans & Ho, 2022). EDM emphasizes that parameterization, preconditioning, loss weighting, schedules, and sampling are distinct design axes (Karras et al., 2022); subsequent work studies time-dependent weighting (Choi et al., 2022; Hang et al., 2023; Kingma & Gao, 2023). Flow matching (Lipman et al., 2023), rectified flow (Liu et al., 2023), and stochastic interpolants (Albergo & Vanden-Eijnden, 2023; Albergo et al., 2023) formulate generation through vector fields. Recent analyses study the interaction between weighting and parameterization (Gagneux

[1] Anonymous code: https://anonymous.4open.science/r/JLT-0482/README.md

Scale	Network predicts	Loss	FID-50K ↓	IS ↑
B/1	velocity v	$\mathcal{L}_v$	6.56	132.12
	clean latent x	$\mathcal{L}_v$	**2.70**	**213.89**
L/1	velocity v	$\mathcal{L}_v$	2.12	236.21
	clean latent x	$\mathcal{L}_v$	**1.47**	**255.28**
H/1	velocity v	$\mathcal{L}_v$	1.60	**327.41**
	clean latent x	$\mathcal{L}_v$	**1.19**	271.96

Table 1: **Parameterization comparison across model scales.** Within each scale, the training setup and velocity objective are matched; only the neural prediction target changes.

et al., 2026) and the dimensionality of different prediction targets (Jin & Wang, 2026). We fix the flow objective and ask which algebraically valid quantity should be exposed directly to the network.

Connection to preconditioning. EDM writes its denoiser as a known skip connection plus a scaled network output, $D_\theta(u;\sigma) = c_{\text{skip}}(\sigma)u + c_{\text{out}}(\sigma)F_\theta(\cdot)$ (Karras et al., 2022). The unregularized JLT readout has the same structural form: $\hat{v} = (1-t)^{-1}f_\theta^x - (1-t)^{-1}z_t$. Within this preconditioning structure, we study the exact target conversion for a linear flow, its placement across the neural/solver interface, and the comparison of direct clean and velocity outputs within a fixed Transformer family. We note that clean reconstruction is the classical objective of denoising autoencoders (Vincent et al., 2008; 2010; Bengio et al., 2013; Alain & Bengio, 2014), while residual denoisers learn the corruption component (Zhang et al., 2017). Natural images have low intrinsic dimensionality (Pope et al., 2021), a property reflected in diffusion scores (Stanczuk et al., 2024). JiT (Li & He, 2026) revisits clean prediction for pixel-space Transformers. Our analysis emphasizes a complementary view: clean prediction places an exact affine transformation outside the learned function, which changes both the effective function class and the geometry of the supervised target.

Latent and representation-space generation. Latent diffusion (Rombach et al., 2022) established generation in pretrained autoencoder spaces, and DiT (Peebles & Xie, 2023), SiT (Ma et al., 2024), Stable Diffusion 3 (Esser et al., 2024), U-ViT (Bao et al., 2023), DiffiT (Hatamizadeh et al., 2024), and PixArt (Chen et al., 2024) develop Transformer-based generators in latent spaces. Other work changes or augments the representation itself: REPA (Yu et al., 2025) and SRA 2 (Wang et al., 2026) use representation alignment, while representation autoencoders (Zheng et al., 2026; Singh et al., 2026) generate in semantic feature spaces. RiT (Zhang et al., 2026) applies clean prediction directly in frozen DINOv2 features (Oquab et al., 2024). DPM-Solver++ (Lu et al., 2025) uses data prediction as a sampling parameterization. In contrast, we freeze a reconstruction-trained VAE and the Transformer family and study the quantity predicted directly during learning, while keeping the vector-field interface to the solver fixed.

3 Same Flow, Different Learning Problems

3.1 The Solver Interface Need Not Be the Neural Interface

Let $x \in \mathbb{R}^D$ be a clean latent from a fixed encoder and $\epsilon \sim \mathcal{N}(0, I)$ be Gaussian noise. We use the linear path

$$z_t = tx + (1-t)\epsilon, \qquad t \in [0,1], \tag{2}$$

whose sample-wise velocity is $v = x - \epsilon$. A direct-velocity model learns

$$f_\theta^v(z_t, t, y) \to v, \tag{3}$$

whereas JLT learns the clean endpoint

$$f_\theta^x(z_t, t, y) \to x, \qquad \hat{v}_\theta^x = \frac{f_\theta^x(z_t, t, y) - z_t}{1-t}. \tag{4}$$

The identities in this section use the unregularized readout for $0 < t < 1$. Both parameterizations expose a velocity interface to an ODE solver. Endpoint regularization in the implementation is discussed separately in Appendix C.

The distinction can be stated without a Gaussian assumption. Under squared error, the Bayes-optimal predictors are conditional means,

$$x^*(z_t, t, y) = \mathbb{E}[x \mid z_t, t, y], \qquad v^*(z_t, t, y) = \mathbb{E}[v \mid z_t, t, y]. \tag{5}$$

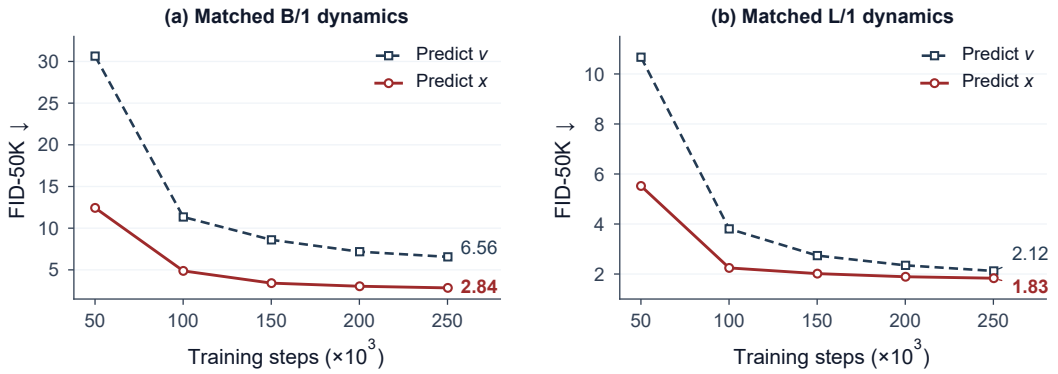

Figure 4: **Matched training dynamics across scales.** (a) Base scale (B/1, 130.5M params): clean-x prediction maintains a 3.72 to 18.20 FID margin over direct velocity across all checkpoints; (b) Large scale (L/1, 458.1M params): clean-x prediction is lower at every measured checkpoint, reaching 1.83 versus 2.12. All checkpoints within each pair are evaluated under the same setting.

Since $v = (x - z_t)/(1 - t)$ sample-wise, conditioning gives the exact identity

$$v^*(z_t, t, y) = \frac{x^*(z_t, t, y) - z_t}{1 - t}. \tag{6}$$

The readout therefore does not approximate an additional unknown function. It implements a known input- and time-dependent transformation of the clean conditional mean.
This changes the effective parameterized family seen by the solver. If $\mathcal{F} = \{f_\theta\}$ denotes the Transformer function class, clean prediction induces

$$\mathcal{V}_x = \left\{ (z, t) \mapsto \frac{f(z, t) - z}{1 - t} : f \in \mathcal{F} \right\}, \tag{7}$$

whereas direct velocity uses $\mathcal{V}_v = \mathcal{F}$. An unrestricted oracle can represent either form, but a fixed finite architecture need not be closed under the affine readout. In architectural terms, the clean parameterization supplies an explicit, time-dependent residual connection from the input z_t to the solver output, rather than asking the Transformer to realize that response internally. Output parameterization therefore decides which computation is learned by the Transformer and which is supplied exactly by the model. Appendix A lists the corresponding conversions among x, ϵ, and v.

3.2 Target Geometry in an Anisotropic Latent Space

We next examine what each neural interface asks the network to regress. Consider a local Gaussian model $x \sim \mathcal{N}(0, \Sigma)$ with independent $\epsilon \sim \mathcal{N}(0, I)$. The eigenvalues of Σ describe directions with strong or weak clean-latent variation.

Proposition 1 (Target covariance and low-variance gain)**.** *For the path in Eq. (2) at $0 < t < 1$, the marginal target covariances are*

$$\mathrm{Cov}(x) = \Sigma, \qquad \mathrm{Cov}(v) = \Sigma + I. \tag{8}$$

For an eigen-direction of Σ with eigenvalue λ_i and $q_i = t^2\lambda_i + (1 - t)^2$, the Bayes estimators are

$$\mathbb{E}[x_i \mid z_i] = \frac{t\lambda_i}{q_i} z_i, \tag{9}$$

$$\mathbb{E}[v_i \mid z_i] = \frac{t\lambda_i - (1 - t)}{q_i} z_i. \tag{10}$$

As $\lambda_i \to 0$, the clean coefficient tends to 0, while the velocity coefficient tends to $-1/(1 - t)$.

The two limits reveal the division of labor created by the readout. Along a direction with little clean-latent variation, the optimal clean predictor attenuates the input. The optimal velocity predictor instead contains a large negative residual response. Equation (6) shows where that response comes from: as the learned clean term vanishes, the fixed $-z_t/(1 - t)$ part of the readout remains. The clean parameterization therefore exposes the structured endpoint to the Transformer and computes the residual response analytically.

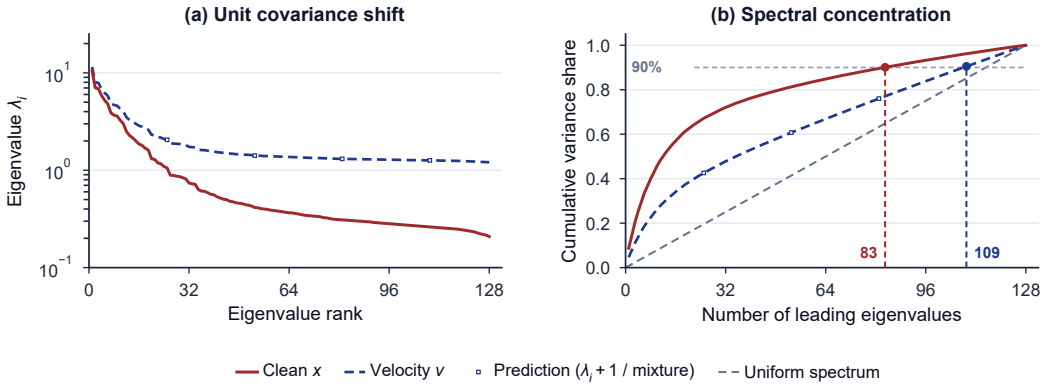

Figure 5: **Velocity spreads target variation across the frozen FLUX latent space.** All 128 channel eigenvalues are estimated over the ImageNet training set. (a) Measured clean and velocity spectra; open squares show the predicted shift $\lambda_i + 1$. (b) Cumulative variance shares. Open squares show the mixture in Eq. (11), and the diagonal is the uniform spectrum. Capturing 90% of target variance requires 83 clean directions and 109 velocity directions. Prediction markers are spaced for readability; all directions enter the calculation.

Figure 3 resolves this behavior across path time and clean variance. Writing the two coefficients as a_x and a_v, the exact readout gives $a_v = (a_x - 1)/(1 - t)$. The clean coefficient is nonnegative, while the velocity coefficient changes sign at $\lambda = (1 - t)/t$ and approaches $-1/(1 - t)$ as the clean variance vanishes.

Spectral concentration. The covariance shift also changes how broadly the target variance is distributed. For descending eigenvalues λ_i and $S = \mathrm{tr}(\Sigma) > 0$, define $C_x(k) = \sum_{i=1}^{k} \lambda_i / S$. The fraction of velocity variance in its leading k directions is

$$C_v(k) = \alpha C_x(k) + (1 - \alpha)\frac{k}{D}, \qquad \alpha = \frac{S}{S + D}. \tag{11}$$

Thus the velocity spectrum mixes the clean spectrum with a uniform spectrum. Since $C_x(k) \geq k/D$, we have $C_v(k) \leq C_x(k)$: adding isotropic noise makes the target variance less concentrated across the clean eigen-directions. This is a statement about target structure rather than numerical conditioning; $\Sigma + I$ can be better conditioned while requiring more directions to capture the same fraction of target variation.

3.3 A Clean Neural Interface

JLT uses a standard latent Transformer. Images are encoded once by a frozen VAE, the latent grid is patchified, and time and class embeddings condition the Transformer blocks through adaptive modulation. The resulting sequence is processed by standard Transformer blocks and unpatchified into $\hat{x}_\theta$. The solver receives only $\hat{v}^x_\theta$ from Eq. (4). The matched direct-velocity model uses the same input, Transformer architecture, conditioning, and solver; only the direct output quantity changes. We use Base, Large, and Huge configurations following the plain JiT family (Li & He, 2026): depths 12, 24, and 32; hidden dimensions 768, 1024, and 1280; and 12, 16, and 16 attention heads. The /1 suffix denotes one latent-grid cell per token. The trainable parameter counts are 130.5M, 458.1M, and 951.3M.

3.4 Training the Same Clean Output with Two Objectives

Our primary comparison uses velocity-space regression. Before endpoint regularization, the clean-output objective is exactly velocity-space MSE for $0 < t < 1$. The clean-output network produces $\hat{x}_\theta$, the fixed readout produces $\hat{v}^x_\theta$, and the loss is

$$\mathcal{L}_v = \mathbb{E}\left[\left\| \frac{\hat{x}_\theta - z_t}{1 - t} - (x - \epsilon) \right\|_2^2\right] \tag{12}$$

$$= \mathbb{E}\left[\frac{\|\hat{x}_\theta - x\|_2^2}{(1 - t)^2}\right]. \tag{13}$$

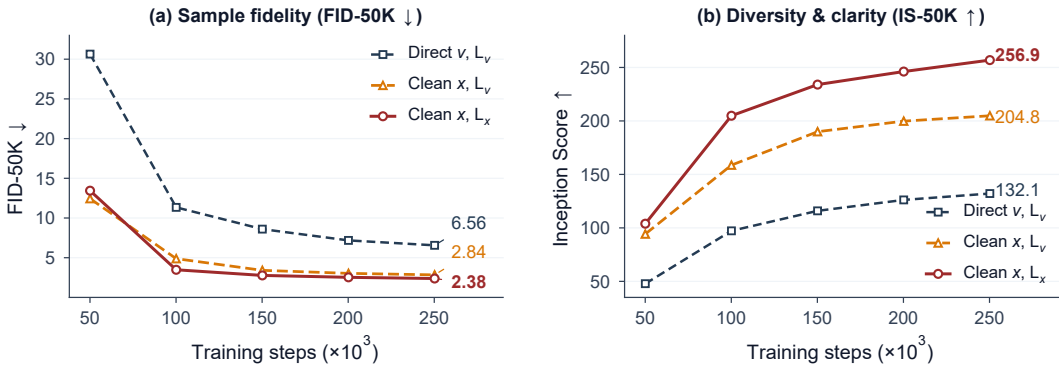

Figure 6: **Loss weighting ablation across training at Base scale.** (a) FID-50K trajectories for direct velocity under $\mathcal{L}_v$ and clean output under $\mathcal{L}_v$ or unweighted $\mathcal{L}_x$; final FIDs are 6.56, 2.84, and 2.38, respectively. (b) Corresponding IS-50K trajectories; final scores are 132.12, 204.83, and 256.88.

The second line shows the clean-error weighting induced by the unregularized velocity readout. The conditional residual variances obey the same scaling (Appendix B). We also train the same clean-output model directly in endpoint space,

$$\mathcal{L}_x = \mathbb{E}\left[\|\hat{x}_\theta - x\|_2^2\right]. \tag{14}$$

This provides a second training interface for the same neural output and the same sampler. JLT denotes the clean-output parameterization, not a particular loss weighting: $\mathcal{L}_v$ and $\mathcal{L}_x$ train the same direct output and expose the same velocity interface at sampling time. In practice, the denominator in Eq. (13) is clipped by $t_\epsilon = 0.05$, and training times follow $\mathrm{logit}(t) \sim \mathcal{N}(-0.8, 0.8^2)$. Appendix C gives the complete setup.

4 Experiments

We organize the experiments along four axes: matched output parameterization, latent covariance, loss weighting with a fixed clean output, and model scale. The matched comparison isolates the neural output while holding the flow objective fixed; the covariance measurement tests the geometric regime identified above; the loss ablation varies weighting without changing the direct output; and the cross-scale comparison across Base, Large, and Huge shows how the parameterization gap evolves with capacity. Together these experiments ask three questions: does changing only the neural output alter learning; does the frozen latent space exhibit the predicted target geometry; and does the clean interface remain effective without time-dependent error weighting?

4.1 Setup

We evaluate class-conditional ImageNet-1K generation at 256×256 (Deng et al., 2009; Russakovsky et al., 2015). Models use deterministic codes from a frozen FLUX.2 VAE (Black Forest Labs, 2026), train for 200 epochs (approximately 250K optimizer steps), and use class-drop probability 0.1. Evaluation generates 50K samples with 50-step Heun integration. We report Fréchet Inception Distance on 50K samples (FID-50K) and Inception Score (IS) (Salimans et al., 2016; Heusel et al., 2017). Appendix C gives the evaluation settings.

4.2 Equivalent Flow, Unequal Learning

Table 1 compares clean and velocity outputs. Within each scale, the models share the frozen encoder, Transformer, time distribution, optimizer, training budget, and velocity objective; their direct neural output differs.

At Base scale, directly predicting the clean latent improves FID-50K from 6.56 to 2.70 (a 3.86 point, or 58.8%, reduction) and raises IS from 132.12 to 213.89. At Large scale, where direct velocity already achieves FID-50K 2.12, clean output reaches 1.47 (a 0.65 point, or 30.7%, reduction) and raises IS from 236.21 to 255.28. At Huge scale (951.3M parameters), clean output attains FID-50K 1.19 and IS 271.96, compared with FID-50K 1.60 for direct velocity.

The distinction is visible throughout optimization. Figure 4 shows the matched B/1 and L/1 trajectories: at Base scale, clean output starts at FID-50K 12.44 versus 30.64 at 50K steps and converges to 2.84 versus 6.56; at Large scale, clean output starts at FID-50K 5.52 versus 10.67 and remains lower

Model	Space	Extra rep.	Params	FID-50K ↓	IS ↑
DiT-XL/2 (Peebles & Xie, 2023)	VAE	–	675M	2.27	278.20
SiT-XL/2 (Ma et al., 2024)	VAE	–	675M	2.06	277.50
REPA-SiT-XL/2 (Yu et al., 2025)	VAE	DINOv2	675M	1.42	305.70
MDTv2-XL/2 (Gao et al., 2024)	VAE	–	675.8M	1.58	314.73
DiffiT (Hatamizadeh et al., 2024)	VAE	–	561M	1.73	276.49
JiT-H/16 (Li & He, 2026)	pixel	–	953M	1.86	303.40
RiT (Zhang et al., 2026)	DINOv2	DINOv2	676M	**1.14**	–
JLT-H/1 ($\mathcal{L}_v$)	VAE	–	951.3M	1.19	271.96

Table 2: **Representative guided ImageNet** 256×256 **models.** Baselines from cited reports; "Extra rep." denotes external self-supervised representations.

at every measured checkpoint, ending at 1.83 versus 2.12. On Qwen-VAE, matched B/1 models show the same FID ordering at the measured checkpoints, reaching FID-50K 6.54 versus 6.85 at 250K steps (Appendix D, Figure 7; Wu et al., 2025).

4.3 Scale Dependence of the Parameterization Gap

Table 1 compares Base, Large, and Huge models trained with the velocity objective $\mathcal{L}_v$. For B/1 and L/1, clean output reduces FID by 3.86 (58.8%) and 0.65 (30.7%), respectively. At H/1, clean output reaches 1.19, compared with 1.60 for direct velocity. The B/1 to L/1 gap narrows as model size increases; the H/1 result shows that clean prediction remains effective at 951.3M parameters. Table 2 situates the 1.19 result alongside contemporary literature.

4.4 Latent Covariance Spectrum: Does the Predicted Regime Hold?

We measure the covariance spectrum of the same frozen FLUX representation used for generation. All 1,281,176 ImageNet training images are encoded into $128 \times 16 \times 16$ latent tensors. Treating each spatial location as an observation, let $X \in \mathbb{R}^{N \times 128}$ contain mean-centered latents after the same frozen channel normalization used in training, and let E contain independent standard Gaussian noise. We estimate

$$\widehat{\Sigma}_x = \frac{1}{N-1} X^\top X, \qquad \widehat{\Sigma}_v = \frac{1}{N-1} (X-E)^\top (X-E). \tag{15}$$

Figure 5 shows the predicted shift directly. The clean channel spectrum ranges from 10.50 to 0.21, a 50.36× span, whereas the velocity spectrum ranges from 11.50 to 1.21, closely following $\lambda_i + 1$. Numerical conditioning improves after the shift: the covariance condition number falls from 50.36 to 9.52. Target concentration moves in the opposite direction, however: the effective rank rises from 35.10 to 78.97 out of 128 channels. The clean representation therefore remains strongly anisotropic after VAE compression, while velocity fills its low-variance tail with unit-scale variation and distributes variance more broadly.

The cumulative spectrum makes the same change directly visible. With clean trace 120.20 over 128 channels, Eq. (11) predicts that the velocity spectrum is a 48.4%/51.6% mixture of the clean and uniform spectra. The measured curve follows this prediction: 90% of clean target variance is captured by 83 leading directions, while velocity requires 109. Thus the key distinction is not numerical conditioning ($\Sigma + I$ is better conditioned), but broader distribution of target variation across directions weakly supported by clean latents.

4.5 Loss Weighting with a Fixed Clean Output

Output parameterization and loss weighting are distinct design choices. As shown in Eq. (13), training clean prediction under $\mathcal{L}_v$ induces a clean-error weighting of $(1-t)^{-2}$ before endpoint clipping. We additionally train the clean-output B/1 model with unweighted clean MSE ($\mathcal{L}_x = \mathbb{E}[\|\hat{x}_\theta - x\|_2^2]$) to examine loss weighting with the same direct output.

Figure 6 compares the convergence trajectories alongside the direct velocity baseline. At 100K steps, clean output under unweighted $\mathcal{L}_x$ reaches FID-50K 3.48 and IS 204.9. At 250K steps, it reaches FID-50K 2.38 and IS 256.88, compared with 2.84 and 204.83 for clean output under $\mathcal{L}_v$. At B/1, clean prediction performs strongly under both velocity-weighted and unweighted clean regression, with the latter reaching FID-50K 2.38.

4.6 Reference Models

Table 2 contextualizes JLT-H/1 alongside representative guided ImageNet 256×256 models. At Huge scale, JLT-H/1 reaches FID-50K 1.19 and IS 271.96. Figure 11 (Appendix F) provides paired qualitative comparisons on B/1.

5 Discussion

Parameterization as a computation boundary. The unregularized clean-to-velocity identity is exact on $0 < t < 1$, but its placement across the model boundary is consequential. Direct velocity embeds the vector-field response inside the Transformer, whereas clean prediction moves the known input residual and gain outside it. For a fixed map T, learning $T \circ f_\theta$ versus unconstrained g_θ yields distinct inductive biases: algebraic equivalence dictates what can be represented, while parameterization dictates what is learned.

Preserving structure in learned representations. Autoencoders compress pixels without yielding isotropic latents. FLUX latents exhibit strong spectral concentration, which clean prediction presents directly to the network; velocity mixes this signal with a uniform noise floor. Beyond altering autoencoders, the prediction target determines whether intrinsic representation geometry is preserved during flow training.

Finite learning versus solver semantics. Both parameterizations expose velocity to the same solver. The FID margin narrows from Base to Large, consistent with the finite-capacity view that externalizing the analytic readout is most beneficial when model capacity is limited. The Huge result shows that the clean-output model remains effective at larger scale.

6 Conclusion

A flow solver requires velocity values, but a generative Transformer need not learn velocity directly. Predicting the clean latent with an exact affine readout preserves target geometry, improves ImageNet generation across Base and Large scales under matched velocity objectives, and scales to 951M parameters, reaching an FID-50K of 1.19 at Huge scale. An ablation at Base scale shows that direct clean regression ($\mathcal{L}_x$) remains highly effective, achieving FID-50K 2.38 without time-dependent error weighting. Algebraically equivalent targets yield unequal learning problems when known linear computation is externalized from the network.

AI Use Statement

Large language models and automated coding assistants assist with language editing, consistency checks, manuscript organization, and LaTeX formatting. The authors retain responsibility for the scientific content and final manuscript.

Ethics Statement

This paper studies prediction parameterization in latent flow generative models using ImageNet-1K and pretrained VAE representations. Experiments focus on class-conditional image generation and the geometry of prediction targets. The authors adhere to the ICLR Code of Ethics.

Reproducibility Statement

We provide the ingredients needed to reproduce the analysis and experiments: (1) Appendices A and B give the target conversions, Gaussian conditional means and residual variances, and the spectral-mixture identity; (2) Section 3 and Appendix C specify the Base/Large/Huge architectures and the optimization recipe, including AdamW hyperparameters, learning rates, warmup, EMA decays, class dropout, and logit-normal time sampling; (3) Section 4 and Appendix C specify the 50-step Heun sampler, evaluation settings, sample count, and covariance-spectrum construction; and (4) the anonymized repository contains source code, model configurations, evaluation scripts, and figure-reproduction code.

REFERENCES

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *Journal of Machine Learning Research*, 2014. URL https://www.jmlr.org/papers/v15/alain14a.html.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *ICLR*, 2023. URL https://arxiv.org/abs/2209.15571.

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023. URL https://arxiv.org/abs/2303.08797.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, 2023. URL https://arxiv.org/abs/2209.12152.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *NeurIPS*, 2013. URL https://arxiv.org/abs/1305.6663.

Black Forest Labs. FLUX.2 Small Decoder. Hugging Face model repository, 2026.

Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-alpha: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *ICLR*, 2024. URL https://arxiv.org/abs/2310.00426.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo J. Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *CVPR*, 2022. URL https://arxiv.org/abs/2204.00227.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024. URL https://arxiv.org/abs/2403.03206.

Anne Gagneux, Ségolène Martin, Rémi Gribonval, and Mathurin Massias. Training flow matching: The role of weighting and parameterization. *arXiv preprint arXiv:2603.06454*, 2026. URL https://arxiv.org/abs/2603.06454.

Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. MDTv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389v2*, 2024. URL https://arxiv.org/abs/2303.14389v2.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via Min-SNR weighting strategy. In *ICCV*, 2023. URL https://arxiv.org/abs/2303.09556.

Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. In *ECCV*, 2024. URL https://arxiv.org/abs/2312.02139.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. URL https://arxiv.org/abs/1706.08500.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org/abs/2207.12598.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. URL https://arxiv.org/abs/2006.11239.

Qing Jin and Chaoyang Wang. Revisiting diffusion model predictions through dimensionality. In *ICML*, 2026. URL https://arxiv.org/abs/2601.21419.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022. URL https://arxiv.org/abs/2206.00364.

Diederik P. Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *NeurIPS*, 2023. URL https://arxiv.org/abs/2303.00848.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *CVPR*, 2026. URL https://arxiv.org/abs/2511.13720.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023. URL https://arxiv.org/abs/2210.02747.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023. URL https://arxiv.org/abs/2209.03003.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 2025. URL https://doi.org/10.1007/s11633-025-1562-4.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *ECCV*, 2024. URL https://arxiv.org/abs/2401.08740.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021. URL https://arxiv.org/abs/2102.09672.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2304.07193.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. URL https://arxiv.org/abs/2212.09748.

Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. In *ICLR*, 2021. URL https://arxiv.org/abs/2104.08894.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. URL https://arxiv.org/abs/2112.10752.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022. URL https://arxiv.org/abs/2202.00512.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016. URL https://arxiv.org/abs/1606.03498.

Jaskirat Singh, Boyang Zheng, Zongze Wu, Richard Zhang, Eli Shechtman, and Saining Xie. Improved baselines with representation autoencoders. *arXiv preprint arXiv:2605.18324*, 2026. URL https://arxiv.org/abs/2605.18324.

Jan Pawel Stanczuk, Georgios Batzolis, Teo Deveney, and Carola-Bibiane Schönlieb. Diffusion models encode the intrinsic dimension of data manifolds. In *ICML*, 2024. URL https://arxiv.org/abs/2212.12611.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. URL https://dl.acm.org/doi/10.1145/1390156.1390294.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 2010. URL https://www.jmlr.org/papers/v11/vincent10a.html.

Mengmeng Wang, Dengyang Jiang, Liuzhuozheng Li, Yucheng Lin, Guojiang Shen, Xiangjie Kong, Yong Liu, Guang Dai, and Jingdong Wang. SRA 2: Variational autoencoder self-representation alignment for efficient diffusion training. In *CVPR*, 2026. URL https://arxiv.org/abs/2601.17830.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025. URL https://arxiv.org/abs/2508.02324.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *ICLR*, 2025. URL https://arxiv.org/abs/2410.06940.

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 2017. URL https://arxiv.org/abs/1608.03981.

Le Zhang, Ning Mang, and Aishwarya Agrawal. Rit: Vanilla diffusion transformers suffice in representation space. *arXiv preprint arXiv:2605.21981*, 2026. URL https://arxiv.org/abs/2605.21981.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *ICLR*, 2026. URL https://arxiv.org/abs/2510.11690.

Appendix

This appendix collects target conversions, theoretical derivations, implementation details, cross-tokenizer generalization analyses, classifier-free guidance sweeps, and uncurated visual samples that support the main paper.

A Target Conversions **14**

B Gaussian Derivation **15**

C Implementation Details **16**

D Cross-Tokenizer Generalization with Qwen-VAE **17**

E Classifier-Free Guidance Sweep **18**

F Qualitative Visual Samples and Comparison **19**

A TARGET CONVERSIONS

For fixed $0 < t < 1$, any one of x, ϵ, and $v = x - \epsilon$ determines the other two from z_t. For clean prediction,

$$\hat{\epsilon}_\theta^{(x)} = \frac{z_t - t\hat{x}_\theta}{1-t}, \qquad \hat{v}_\theta^{(x)} = \frac{\hat{x}_\theta - z_t}{1-t}. \tag{16}$$

For noise prediction,

$$\hat{x}_\theta^{(\epsilon)} = \frac{z_t - (1-t)\hat{\epsilon}_\theta}{t}, \qquad \hat{v}_\theta^{(\epsilon)} = \frac{z_t - \hat{\epsilon}_\theta}{t}. \tag{17}$$

For velocity prediction,

$$\hat{x}_\theta^{(v)} = z_t + (1-t)\hat{v}_\theta, \qquad \hat{\epsilon}_\theta^{(v)} = z_t - t\hat{v}_\theta. \tag{18}$$

These identities also expose error scaling. Let $e_x = \hat{x}_\theta - x$, $e_\epsilon = \hat{\epsilon}_\theta - \epsilon$, and $e_v = \hat{v}_\theta - v$. Then

$$\hat{v}_\theta^{(x)} - v = \frac{e_x}{1-t}, \qquad \hat{\epsilon}_\theta^{(x)} - \epsilon = -\frac{t}{1-t}e_x, \tag{19}$$

$$\hat{x}_\theta^{(\epsilon)} - x = -\frac{1-t}{t}e_\epsilon, \qquad \hat{v}_\theta^{(\epsilon)} - v = -\frac{1}{t}e_\epsilon, \tag{20}$$

$$\hat{x}_\theta^{(v)} - x = (1-t)e_v, \qquad \hat{\epsilon}_\theta^{(v)} - \epsilon = -te_v. \tag{21}$$

The targets are convertible, but equal MSE in one target is not equal MSE in another.

B GAUSSIAN DERIVATION

Let $x_i \sim \mathcal{N}(0, \lambda_i)$ and $\epsilon_i \sim \mathcal{N}(0, 1)$ be independent in an eigenbasis of Σ. With $z_i = tx_i + (1-t)\epsilon_i$ and $q_i = t^2\lambda_i + (1-t)^2$, joint Gaussian conditioning gives

$$\mathrm{Var}(a \mid z) = \mathrm{Var}(a) - \frac{\mathrm{Cov}(a, z)^2}{\mathrm{Var}(z)}. \tag{22}$$

Using $\mathrm{Cov}(x_i, z_i) = t\lambda_i$, $\mathrm{Cov}(\epsilon_i, z_i) = 1 - t$, and $\mathrm{Cov}(v_i, z_i) = t\lambda_i - (1-t)$ yields

$$\mathrm{Var}(x_i \mid z_i) = \frac{\lambda_i(1-t)^2}{q_i}, \tag{23}$$

$$\mathrm{Var}(\epsilon_i \mid z_i) = \frac{t^2\lambda_i}{q_i}, \tag{24}$$

$$\mathrm{Var}(v_i \mid z_i) = \frac{\lambda_i}{q_i}. \tag{25}$$

Therefore $\mathrm{Var}(v_i \mid z_i) = \mathrm{Var}(x_i \mid z_i)/(1-t)^2$. Summing across directions gives local Bayes residual risks

$$\mathcal{R}_x(t) = \sum_i \frac{\lambda_i(1-t)^2}{t^2\lambda_i + (1-t)^2}, \tag{26}$$

$$\mathcal{R}_v(t) = \sum_i \frac{\lambda_i}{t^2\lambda_i + (1-t)^2}. \tag{27}$$

Together with Proposition 1, these expressions quantify how the same sample-wise target conversion changes conditional scale and regression gain under the local Gaussian model.

Spectral-mixture identity. Adding I preserves the covariance eigenvectors and shifts each eigenvalue by one. The leading-k velocity variance is therefore $\sum_{i=1}^{k} \lambda_i + k$, and total velocity variance is $S + D$. Dividing gives Eq. (11). For descending eigenvalues, the mean of the first k values is at least the mean of all D values, so $C_x(k) \geq k/D$. Substitution yields $C_v(k) \leq C_x(k)$, with equality for an isotropic spectrum.

C Implementation Details

Representation. Images are encoded deterministically with the frozen FLUX.2 VAE. The implementation patchifies the VAE output into a 128-channel, 16×16 latent grid and normalizes each channel using frozen running statistics. The decoder is used only after ODE integration.

Architecture. Table 3 gives the model family. Each block uses self-attention, a swish-gated linear unit (SwiGLU) feed-forward network, root mean square (RMS) normalization, rotary position embeddings, and adaptive time/class modulation. The output projection reconstructs the latent grid. Attention and projection dropout rates are zero.

Model	Depth	Width	Heads	Params
JLT-B/1	12	768	12	130.5M
JLT-L/1	24	1024	16	458.1M
JLT-H/1	32	1280	16	951.3M

Table 3: **JLT architecture family.** Parameter counts include the trainable generator and exclude the frozen VAE.

Optimization. We use AdamW (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, zero weight decay, base learning rate 5×10^{-5}, and effective learning rate 2×10^{-4} after batch scaling. Training lasts 200 epochs with five warmup epochs. Exponential moving averages (EMA) use decay 0.9999 and 0.9996. The class condition is independently dropped with probability 0.1.

Time and sampling. The exact conversion identities above concern the unregularized field on $0 < t < 1$. To distinguish endpoint conventions, define $d_\epsilon(t) = \max(1 - t, t_\epsilon)$. If both the clean-output prediction and its training target use this denominator, their residual is

$$\frac{\hat{x} - z_t}{d_\epsilon(t)} - \frac{x - z_t}{d_\epsilon(t)} = \frac{\hat{x} - x}{d_\epsilon(t)}. \tag{28}$$

The resulting clean-error weight is $d_\epsilon(t)^{-2}$. A sampling readout with the same denominator equals the unregularized field multiplied by $(1 - t)/d_\epsilon(t)$, with exact agreement for $t \leq 1 - t_\epsilon$. Training samples $\mathrm{logit}(t) \sim \mathcal{N}(-0.8, 0.8^2)$. Division by $1 - t$ is clipped at $t_\epsilon = 0.05$. Evaluation uses 50 Heun steps, 50K generated images, classifier-free guidance (Ho & Salimans, 2022), and guidance interval $[0.1, 1]$. Training trajectories and qualitative samples use the same sampling settings within each comparison. Appendix E reports guidance sweeps.

D CROSS-TOKENIZER GENERALIZATION WITH QWEN-VAE

We repeat the matched B/1 comparison on the pretrained Qwen-VAE representation (Wu et al., 2025) to examine transfer across autoencoder representations. Both models use 200 epochs (250K steps), velocity MSE ($\mathcal{L}_v$), the same logit-normal time distribution, and the same sampling settings.

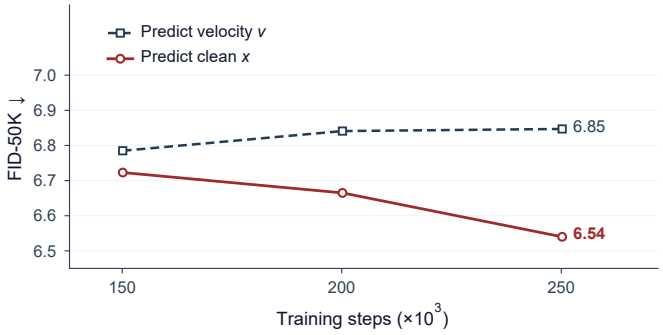

Figure 7: **Matched training dynamics on Qwen-VAE.** FID-50K of matched B/1 models on Qwen-VAE at 150K, 200K, and 250K steps. Clean-x prediction has lower FID at all three measured checkpoints, ending at 6.54 versus 6.85.

Figure 7 and Table 4 show the convergence behavior and final metrics. At the measured checkpoints from 150K to 250K steps, clean-x prediction has lower FID-50K than velocity prediction. The margin grows from 6.72 vs. 6.79 at 150K steps to 6.54 vs. 6.85 at 250K steps (Table 4). The same FID ordering is observed on Qwen-VAE.

Model	Representation	Direct Output	Steps	FID-50K ↓
JLT-B/1	Qwen-VAE	clean latent x	150K	6.72
JLT-B/1	Qwen-VAE	velocity v	150K	6.79
JLT-B/1	Qwen-VAE	clean latent x	200K	6.67
JLT-B/1	Qwen-VAE	velocity v	200K	6.84
JLT-B/1	Qwen-VAE	clean latent x	250K	**6.54**
JLT-B/1	Qwen-VAE	velocity v	250K	6.85

Table 4: **Matched parameterization comparison on Qwen-VAE.** Both models are B/1 configurations trained under the same velocity MSE objective ($\mathcal{L}_v$) and evaluated under the same sampling settings across convergence checkpoints. Clean prediction achieves lower FID at every evaluated stage, ending at 6.54 vs. 6.85 at 250K steps.

E CLASSIFIER-FREE GUIDANCE SWEEP

JLT-B/1 guidance sweep. We evaluate the 200-epoch JLT-B/1 clean-output model across CFG scales from 1.0 to 3.0 in increments of 0.1. The lowest observed FID-50K is 2.70 at CFG 3.0, with IS 213.89. Figure 8 shows the full sweep.

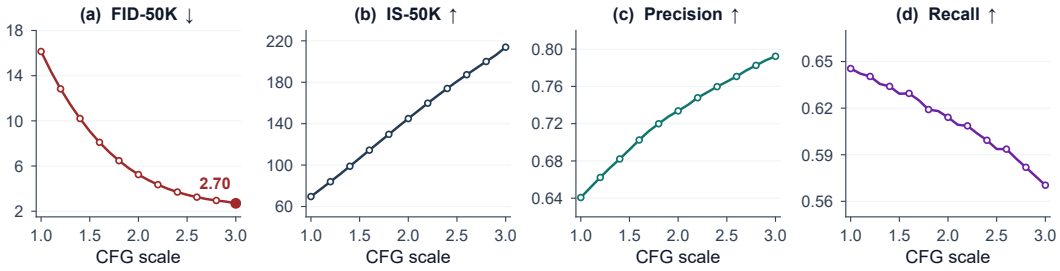

Figure 8: **Classifier-free guidance sweep for JLT-B/1.** FID, IS, precision, and recall across the evaluated CFG grid.

JLT-L/1 guidance sweep. We evaluate the 200-epoch JLT-L/1 clean-output model across CFG scales from 1.0 to 3.0 in increments of 0.1. The lowest observed FID-50K is 1.47 at CFG 2.3, with IS 255.28. Figure 9 shows the full sweep.

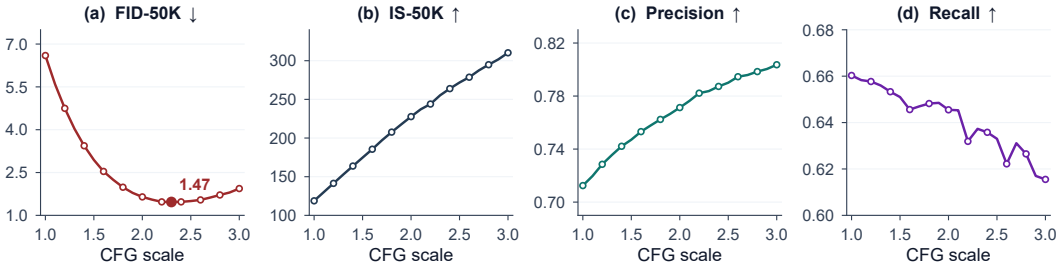

Figure 9: **Classifier-free guidance sweep for JLT-L/1.** FID, IS, precision, and recall across the evaluated CFG grid; clean-x achieves an lowest observed FID-50K of 1.47 at CFG 2.3.

JLT-H/1 guidance sweep. We evaluate the 200-epoch JLT-H/1 clean-output model across CFG scales from 1.0 to 3.0 in increments of 0.1. The lowest observed FID-50K is 1.193 at CFG 2.2, with IS 271.96. Figure 10 shows the full sweep.

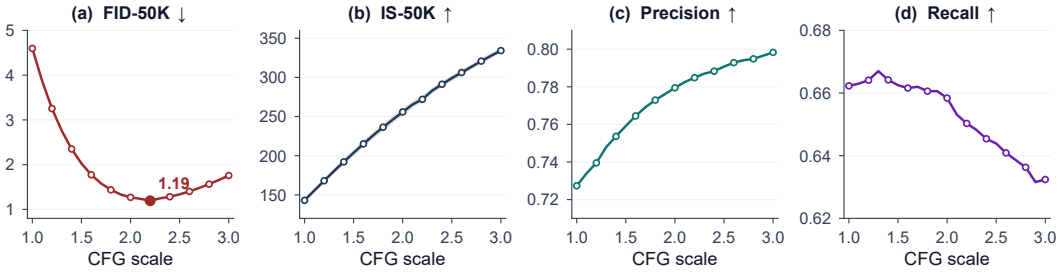

Figure 10: **Classifier-free guidance sweep for JLT-H/1.** FID, IS, precision, and recall across the evaluated CFG grid; the IS band shows one standard deviation.

F Qualitative Visual Samples and Comparison

Figure 11 displays paired class-conditional ImageNet 256×256 generations produced by matched JLT-B/1 models with clean-x and direct velocity-v prediction under the same sampling settings. The paired samples illustrate differences in structure and sharpness across the displayed animal classes.



Figure 11: **Additional paired qualitative comparison on ImageNet** 256×256 **under matched B/1 scale.** Generations from matched JLT-B/1 models trained for 200 epochs and sampled under the same settings. Left sub-columns (x-pred, green) show clean prediction; right sub-columns (v-pred, red) show direct velocity prediction across 10 paired class-conditional samples.