TRAINING FLOW MATCHING: THE ROLE OF WEIGHTING AND PARAMETERIZATION

Anne Gagneux[*1], **Ségolène Martin**[*23], **Rémi Gribonval**[3] **& Mathurin Massias**[3]
[1] ENS de Lyon, CNRS, Université Claude Bernard Lyon 1, Inria, LIP, UMR 5668, France
[2] Technische Universität Berlin, Germany
[3] Inria, ENS de Lyon, CNRS, Université Claude Bernard Lyon 1, LIP, UMR 5668, France

ABSTRACT

We study the training objectives of denoising-based generative models, with a particular focus on loss weighting and output parameterization, including noise-, clean image-, and velocity-based formulations. Through a systematic numerical study, we analyze how these training choices interact with the intrinsic dimensionality of the data manifold, model architecture, and dataset size. Our experiments span synthetic datasets with controlled geometry as well as image data, and compare training objectives using quantitative metrics for denoising accuracy (PSNR across noise levels) and generative quality (FID). Rather than proposing a new method, our goal is to disentangle the various factors that matter when training a flow matching model, in order to provide practical insights on design choices.

1 INTRODUCTION

Flow matching (FM; Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023; Liu et al., 2023) and diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) are the state-of-the-art generative methods. Despite widespread adoption, many fundamental questions remain open, most notably regarding our understanding of why these models perform so well in practice. In this work, we seek to shed new light on some critical design choices: *Which loss weightings and parameterizations should be preferred during training, and how do these choices depend on the architecture and data regime?*

To answer this, we unify different target predictions under a *common weighted denoising formulation*. This viewpoint allows us to directly relate denoising performance, measured at different noise levels, to generative performance. We make the following contributions:

1. *For weights (Section 4):* We provide statistical theoretical insights into why the weights corresponding to flow-matching and signal-to-noise ratio perform robustly across a wide range of settings. To the best of our knowledge, this offers **the first principled explanation for this empirical observation.**
2. *For parametrizations (Section 5):* Recent works show superior performance of denoiser parameterization (parametrization predicting the clean image), and motivate it by a manifold assumption. We find that while this behavior indeed holds in certain settings, velocity parameterization still remains more effective across many scenarios. In particular we show that, **more than the manifold assumption on its own, the locality induced by the network architecture as well as the data regime have a critical influence** when deciding between denoiser or velocity parametrizations.

2 BACKGROUND AND RELATED WORKS

Flow matching and diffusion are equivalent in the case of Gaussian source distribution (Gao et al., 2025): we use the flow matching conventions, but the results are easily translated for diffusion.

Background on flow matching The generative process is defined over time $t \in [0, 1]$, with an initial sample $x_0 \sim p_0$ and a target sample $x_1 \sim p_1$. To connect with the concept of denoisers in the

[*]Equal contribution.

sequel, we further assume that the latent distribution is standard Gaussian: $p_0 = \mathcal{N}(0, \mathrm{I}_d)$, and we work in the setting where the coupling $p_{(x_0,x_1)}$ is the product coupling $p_0 \otimes p_1$. In flow matching, generation of new samples is performed by solving on the differential equation $\dot{x}(t) = v(x(t), t)$ from $t = 0$ to 1, using as initial condition $x(0) = x_0 \sim p_0$. The function $v : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$ is called the *velocity*, and the generated sample is simply the ODE solution at time $t = 1$, namely $x(1)$; for an appropriate velocity, it should behave like a sample from p_1. In practice, the velocity is parametrized by a neural network v_θ and learned by solving:

$$\min_\theta \mathbb{E}_{t\sim\mathcal{U}[0,1],\, x_0\sim p_0,\, x_1\sim p_1} \left[\|v_\theta(x_t, t) - (x_1 - x_0)\|^2\right], \tag{1}$$

where $x_t := (1 - t)x_0 + tx_1$ is the linear interpolation between x_0 and x_1. It is well-known that the solution $v^\star$ to this problem (over all measurable functions) is given by the conditional expectation, $v^\star(x, t) = \mathbb{E}[x_1 - x_0 \mid x_t = x, t]$.

Two important variations can be made in equation 1: first, it is possible to use a time-dependent weight in the loss to prioritize some noise levels, thus improving performance in some cases. Second, a quick computation shows that $v^\star(x, t)$ also equals $\frac{\mathbb{E}[x_1|x_t=x]-x}{1-t}$ and $\frac{x-\mathbb{E}[x_0|x_t=x]}{t}$. Therefore, having a network v_θ learn $v^\star$ (so-called v-prediction) is theoretically equivalent to having a network x_θ learn $\mathbb{E}[x_1|x_t = x]$ (denoising, aka x-prediction) or a network ε_θ learn $\mathbb{E}[x_0|x_t = x]$ (noise prediction, aka ε-prediction). We review these approaches below.

Loss weightings Ever since the first successes of diffusion, a large body of work has studied the choice of training objectives and loss weightings across time (Hang et al., 2023; Kingma & Gao, 2023). We briefly review this literature *with the notations of the diffusion framework, which we adopt in this paragraph for simplicity*. We consider the variance-preserving diffusion process, where noisy samples are generated as $x_t = \alpha_t x_0 + \sigma_t \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, I_d)$, $\alpha_t^2 + \sigma_t^2 = 1$, and time evolves from $t = T$ (pure noise) to $t = 0$ (clean data). Regarding loss weighting, the original diffusion formulation of Ho et al. (2020) employs an unweighted ε-prediction loss, $\mathbb{E}\left[\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2\right]$. Viewing the problem as denoising, with the change of variable $D_\theta(x, t) = (\mathrm{Id} - \sigma_t \varepsilon^\theta(x, t))/\alpha_t$, this is equivalent to minimizing a denoising loss $\|x_0 - D_\theta(x_t, t)\|^2$ but with an additional loss weighting equal to the SNR, α_t^2/σ_t^2, prioritizing denoising of almost clean images. Several works have proposed alternative weightings to emphasize specific noise regimes. For instance, in x-prediction, Salimans & Ho (2022) suggest the weighting $\max(\alpha_t^2/\sigma_t^2, 1)$, while Yu et al. (2024) use α_t/σ_t. Compared to the aforementioned SNR weighting, both assign increased importance to large noise levels, which are argued to be more difficult and critical for error propagation during sampling. Interpreting diffusion training as a multitask optimization problem, Hang et al. (2023) observe gradient conflicts between noise levels and, with a similar objective, propose clipping the weighting as $\min(\alpha_t^2/\sigma_t^2, \gamma)$ to avoid over-emphasizing low-noise, easy denoising tasks. In contrast, Choi et al. (2022) introduce the P2 weighting, which emphasizes intermediate noise levels under the hypothesis that perceptually relevant features emerge during this "content" phase. We emphasize that so far, these design choices rely on empirical observation and heuristics, and a consensus has not been reached.

Unifying perspectives Several works have proposed unifying views on loss weightings in diffusion and flow-matching models. In particular, Kingma & Gao (2023) reinterpret a wide range of objectives and weightings, including flow matching, as differently weighted ELBO formulations, while Kumar et al. (2025); Gao et al. (2025) systematically relate common parameterizations (noise, score, velocity) to the loss weightings they induce. While these works provide valuable theoretical unification, they do not investigate *why and when* different weighting choices lead to substantially different empirical performance. In Section 4, we address this gap through a denoising-based analysis that directly connects loss weighting to both denoising and generation performance.

Data-prediction versus velocity-prediction In contrast with flow matching, which traditionally uses v-prediction, most diffusion implementations followed Ho et al. (2020) and train networks to predict the noise ε. Nevertheless, v-prediction or x-prediction have also been explored (Salimans & Ho, 2022; Hang et al., 2023). In particular, a recent work by Li & He (2025) advocates that, real data lying on a low dimensional manifold (De Bortoli, 2022), x-prediction should be preferred, as the network's task is thus made simpler. Subsequent work by Jin & Wang (2026) provably show on toy linear models the benefits of x-prediction when the data lives in a low dimensional subset of a large space. In Section 5, we revisit this take and show the importance of other factors beyond the manifold assumption.

Table 1: Summary of the denoising weights (left) and parametrization classes (right). Design choices are usually paired by row (e.g. w^t_{den} with $\mathcal{C}_{\text{den}}$), but any combination is possible.

Weights	**Parametrization classes $\mathcal{C}$**
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{den}}$: $D(x,t) = N^\theta(x,t)$
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$: $D(x,t) = x + (1-t)N^\theta(x,t)$
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{noise}}$: $D(x,t) = (x - (1-t)N^\theta(x,t))/t$
$w^t_{\text{classic}} = \mathbf{1}_{[(1+\sigma_{\max})^{-1},1]}(t) \cdot t^{-2}$	

3 Generation as denoising

Any choice of target between the clean image, the noise and the velocity naturally induces a choice of parametrization (i.e., what the network outputs) together with a regression loss. However, any combination of parameterization and loss is in fact possible (e.g. having the noise as the output of the neural network N^θ, but regressing against the clean image in the loss: $\mathbb{E}[\|x_1 - \frac{x_t-(1-t)N^\theta(x_t,t)}{t}\|]$); see for example (Gao et al., 2025, Sec. "Training") or Li & He (2025, Table 1).

A common ground for comparison To express all losses in a common framework for easier comparison, we lay down what these choices lead to from the point of view of the denoising problem. We thus write all instances as:

$$\underset{D\in\mathcal{C}}{\text{minimize}}\ \mathcal{L}(D) = \mathbb{E}_{\substack{t\sim\mathcal{U}[0,1] \\ x_0\sim\mathcal{N}(0,I_d) \\ x_1\sim p_{\text{data}}}} \left[w^t \|D(x_t,t) - x_1\|^2\right]. \tag{2}$$

where $\mathcal{C}$ is the class of learnable functions the denoiser belongs to, which is determined by the choice of targeted network output*, and $\mathcal{L}(D)$ is the training loss. In the end, all possible choices only end up differing by the value of the weighting w^t they induce and the parametrization class $\mathcal{C}$ (explicit values are summarized in Table 1). Even though all versions of the loss $\mathcal{L}$ *share the same minimizer* regardless of the choice of the weighting scheme w^t, restricting minimization to a parametric and learnable function class $\mathcal{C}$ leads to different solutions in practice.

Parametrization classes The formulas are obtained by noticing that the ideal denoiser is $\mathbb{E}[x_1|x_t = x]$, which is also equal to $x+(1-t)\mathbb{E}[x_1-x_0|x_t = x]$ and $(x - (1-t)\mathbb{E}[x_0|x_t = x])/t$. Translating these two relationships to trained velocities, denoisers or noise predictors yields the classes $\mathcal{C}_{\text{den}}$, $\mathcal{C}_{\text{vel}}$ and $\mathcal{C}_{\text{noise}}$ respectively:

$$\mathcal{C}_{\text{den}} = \left\{D : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d \mid D = N^\theta,\ \theta \in \Theta\right\}, \tag{3}$$

$$\mathcal{C}_{\text{vel}} = \left\{D : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d \mid D(\cdot,t) = \text{Id} + (1-t)N^\theta(\cdot,t),\ \theta \in \Theta\right\}, \tag{4}$$

$$\mathcal{C}_{\text{noise}} = \{D : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d \mid D(\cdot,t) = \tfrac{1}{t}\left(\text{Id} - (1-t)N^\theta(\cdot,t)\right),\ \theta \in \Theta\}, \tag{5}$$

where N^θ is any neural network with trainable parameters $\theta \in \Theta$.

Losses and induced weightings As far as the regression target is concerned:

- Usual least-squares regression on the clean image x_1 corresponds to equation 2 for $w^t = 1$.
- For least-squares on the velocity v: using $v(x,t) = \frac{D(x,t)-x}{1-t}$ together with $x_1 - x_0 = \frac{x_1-x_t}{1-t}$, one can rewrite the loss under the form equation 2 for $w^t = 1/(1-t)^2$.
- For least-squares on the noise x_0, using that the noise estimate equals $\frac{x-tD(x,t)}{1-t}$ while the noise is $x_0 = \frac{x_t-tx_1}{1-t}$ yields the weight $w^t = \frac{t^2}{(1-t)^2}$, which is the SNR.

Beyond these three choices, we also study *classical denoisers*, i.e. as they were used prior to the generative era: parameterized by a noise level σ and trained on inputs $x_\sigma = x_1 + \sigma x_0$ (not

*we emphasize that D is not necessarily the network output: we only *rewrite* the problem in terms of denoiser, but the network N^θ may output an estimate of v, x_1 or x_0

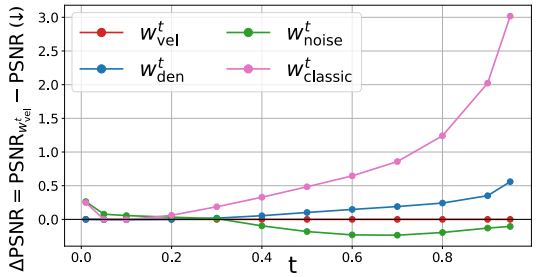
x_t). Such denoisers $\tilde{D}$ are usually trained on noise levels ranging from 0 to $\sigma_{\max}$, by minimizing $\mathcal{L}(\tilde{D}) = \mathbb{E}_{\sigma\sim\mathcal{U}([0,\sigma_{\max}])}\left[\|\tilde{D}(x_\sigma,\sigma) - x_1\|^2\right]$ which is equivalent to using the weighting[†] $w^t_{\text{classic}} = \mathbf{1}_{[(1+\sigma_{\max})^{-1},1]}(t)/t^2$.

Decoupling weightings and parametrizations Usually, weighting w^t_{den} is coupled with parametrization $\mathcal{C}_{\text{den}}$, and similarly for the other pairs. However, any combination is possible, and we will indeed show next that such decoupling is desirable. For example, with parametrization $\mathcal{C}_{\text{den}}$, it is preferable to use w^t_{vel} or w^t_{noise}, which we explain statistically in Section 4.

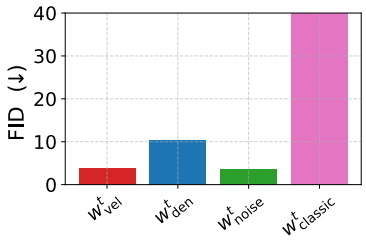
4 Evaluating the impact of weightings

To disentangle the respective roles of parameterization classes from that of weightings, as a first investigation, we evaluate several time-weighting strategies to train Flow Matching models. We focus on the parametrization $\mathcal{C}_{\text{vel}}$, which is the most commonly used in practice. Experiments are conducted on CIFAR-10 (32×32, Krizhevsky & Hinton, 2009) and CelebA-64 (64×64, Yang et al., 2015). All models share the same U-Net architecture and training hyperparameters, taken from the standard FM setup (full details are provided in Appendix A). To isolate the effect of weighting alone, *the time t is uniformly sampled during training*: note that Li & He (2025), when comparing losses, opt for a non-uniform sampling $\text{logit}(t) \sim \mathcal{N}(0,1)$ which further modifies the weighting in practice.

Metrics For generative models it is customary to report the Fréchet Inception Distance (FID, Heusel et al., 2017), which measures the similarity between generated and real image distributions in a feature space. However, evaluating the quality of a denoiser at each noise level provides a complementary perspective. We therefore introduce Peak Signal-to-Noise Ratio (PSNR) curves: at a fixed time t we compute the peak signal-to-noise ratio between the denoiser output $D(x_t,t)$ and clean images x_1. Higher PSNR indicates more accurate denoising. Unlike FID, PSNR evaluation is fast, does not require natural colour images, and pinpoints the noise levels at which a model performs poorly. Moreover, PSNR reveals overfitting: a model that memorises the training data might achieve excellent FID but poor PSNR because its denoiser output collapses to training examples.



(a) Difference in PSNR (lower is better) between the reference w^t_{vel}, and various models, computed on 1000 test images. The parametrization class is here set to $\mathcal{C}_{\text{vel}}$.



(b) FID on 50k train images (lower is better).

Figure 1: PSNR and FID for the different losses, CIFAR-10. Models that reach the highest PSNR (low difference in PSNR compared to standard FM, w^t_{vel}) also reach the lowest FID.

Numerical results Figure 1 reports both denoising quality (PSNR) and generation quality (FID) for the different weightings on CIFAR-10 (see Figure 7 for CelebA-64). A first notable observation is that **denoising performance and sample quality are strongly correlated**: models achieving better PSNR also tend to obtain better FID. The **best results are consistently obtained with the SNR weighting** $w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$, which achieves both the highest PSNR and the lowest FID. The standard

[†]in other words, classical denoisers cannot handle very low SNR regimes except if trained with unbounded noise levels. In practice, we set $\sigma_{\max} = 19$ so that $t_{\min} = 1/(1+\sigma_{\max}) = 0.05$ – in traditional denoising, models are usually trained with noise level at most $\sigma = 100/255 \simeq 0.4$

Flow Matching weighting $w^t_{\text{vel}} = \frac{1}{(1-t)^2}$ comes second, with performance very close to the SNR one. Interestingly, these weightings prioritize the low-noise regime (t close to 1), where denoising might appear easier, yet it remains crucial for overall performance. Moreover, we investigated other exploding weightings of the form $w^t = (1-t)^{-p}$ with $p \in \{1, 3\}$. These variants perform worse than the standard choice $p = 2$ (see Appendix B) arguing for optimality of the quadratic scaling of w^t_{vel} that we further investigate statistically below in Section 4.

We also examine the classical weighting w^t_{classic}, which is implicitly adopted in much of the imaging and inverse problems literature. Our experiment indicates that this choice is suboptimal when training over a wide range of noise levels, even for pure denoising tasks. We provide the same comparison using the alternative parametrizations $\mathcal{C}_{\text{den}}$ and $\mathcal{C}_{\text{noise}}$ in Appendix B, with identical conclusions.

Understanding optimal weightings near $t = 1$ We now leverage the denoising viewpoint to shed light on why the two best performing weightings, Flow Matching and SNR, behave like $1/(1-t)^2$ as $t \to 1$. To do so, we draw a conceptual link with classical results on inverse-variance weighting in heteroscedastic regression and maximum likelihood estimation (Shalizi, 2013; Schick, 1997; Aitken, 1936).

In a regression setup, we consider the rescaled variable $y_t := \frac{x_t}{t} = x_1 + \frac{1-t}{t} x_0$, i.e. a noisy observation of x_1. The learning objective is therefore to estimate the conditional mean denoiser $\mathbb{E}[x_1 | y_t = y]$ with a neural network $f_\theta(y, t)$. Since the noise level depends on t, this is an heteroscedastic regression problem, and a natural approach is to model the conditional distribution $p(x_1 | y_t = y)$ and estimate $\mathbb{E}[x_1 | y_t = y]$ by maximum likelihood.

When $t \to 1$, the corruption level goes to 0, so the conditional distribution $p(x_1 | y_t = y)$ concentrates around its mean. In this regime, we approximate the posterior by a Gaussian distribution: $p(x_1 | y_t = y) = \mathcal{N}\big(\mathbb{E}[x_1 | y_t = y], \Sigma(t)\big)$, with a covariance $\Sigma(t) \to 0$ as $t \to 1$. Replacing the unknown conditional mean by its neural approximation yields the model $p_\theta(x_1 | y, t) \approx \mathcal{N}\big(x_1; f_\theta(y, t), \Sigma(t)\big)$. A natural way to estimate f_θ is then through maximum likelihood. Up to an additive constant, at a fixed time t, the negative log-likelihood equals $\frac{1}{2}\left(x_1 - f_\theta(y,t)\right)^\top \Sigma(t)^{-1} (x_1 - f_\theta(y,t))$. Since the same parameters θ are shared across all t, training requires averaging this objective over both t and the data distribution. This leads to the weighted regression loss

$$\mathbb{E}\Big[(x_1 - f_\theta(y_t, t))^\top \Sigma(t)^{-1} (x_1 - f_\theta(y_t, t))\Big]. \tag{6}$$

Therefore, maximum likelihood naturally prescribes an *inverse-covariance weighting*: time steps where the conditional variance is small (in particular near $t = 1$) should receive larger weight. This provides a statistical justification for weightings that diverge as $t \to 1$. This is indeed the case of $w(t) \propto (1-t)^{-2}$, and we now specifically investigate the optimality of this scaling on a toy model.

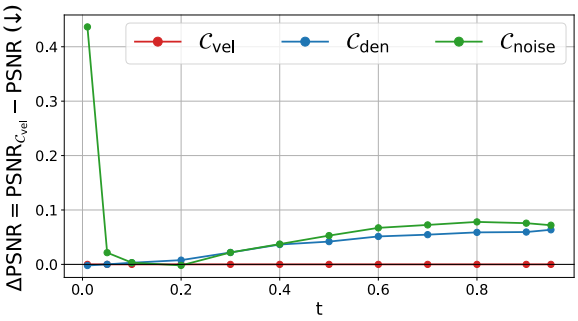
Finally, to make this reasoning explicit, we consider the analytically tractable setting where the data is Gaussian: in addition to $x_0 \sim \mathcal{N}(0, I)$, $x_1 \sim \mathcal{N}(0, \tau^2 I)$. In this linear–Gaussian case, the posterior admits by Bayes formula the closed form:

$$x_1 | (y_t = y, t) \sim \mathcal{N}\Big(\frac{\tau^2}{\tau^2 + \frac{(1-t)^2}{t^2}}\, y,\ \sigma^2_{\text{post}}(t) I\Big), \quad \text{with } \sigma_{\text{post}}(t) = \left(\frac{1}{\tau^2} + \frac{t^2}{(1-t)^2}\right)^{-1} \tag{7}$$

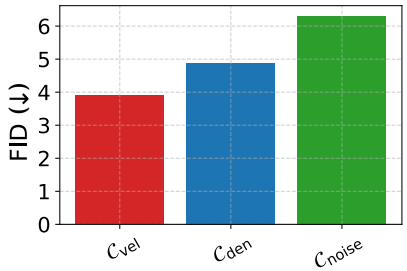
From equation 6, the optimal weighting is therefore $\sigma^{-2}_{\text{post}}(t) = \frac{1}{\tau^2} + \frac{t^2}{(1-t)^2}$. In particular, as $t \to 1$, we recover the growth in $\frac{1}{(1-t)^2}$ which matches both SNR and standard Flow Matching weighting. Thus, in this model, the empirically successful choice $w(t) \sim (1-t)^{-2}$ **emerges directly from inverse-variance weighting under a maximum-likelihood interpretation**.

5 Evaluating the impact of parametrizations

We now turn to the role of parametrization and first assess the denoising and generation performance for each choice of parametrization, as done for the weightings in Figure 1; the weighting scheme is fixed to w^t_{vel} (results for the other weighting schemes, as well as additional results on CelebA-64, are in Appendix B). Figure 2 demonstrates that **the velocity parametrization $\mathcal{C}_{\text{vel}}$ consistently**



(a) Difference in PSNR (lower is better) between the reference $\mathcal{C}_{\text{vel}}$, and other parametrizations, computed on 1000 test images. The weight is set to w^t_{vel}.



(b) FID on 50k train images (lower is better).

Figure 2: PSNR and FID for the different parametrizations, CIFAR-10. Models that reach the highest PSNR (low difference in PSNR compared to standard FM) also reach the lowest FID.

gives better performance, regarding both generation and denoising at every noise level (i.e., at every time t). The noise parametrization fails critically at early times (high noise levels), which can be understood from a simple geometric consideration: when $t \to 0$, the denoiser should output the mean of the data distribution. This behavior is prevented by the factor $1/t$ in $\mathcal{C}_{\text{noise}}$, which explodes at large noise levels. In addition, these experiments show that learning only the noise ($\mathcal{C}_{\text{noise}}$) or the clean image ($\mathcal{C}_{\text{den}}$) results in a degraded *denoising* performance.

Is the velocity parametrization really optimal? Our first experiments consistently favor velocity parameterization on both CIFAR-10 and CelebA-64. This is in agreement with early FM results by Lipman et al. (2023, Table 1), who already put forward that velocity prediction outperformed noise prediction and score prediction. However, Li & He (2025) recently reported conclusions that appear to contradict these findings, showing that the denoiser parameterization $\mathcal{C}_{\text{den}}$ significantly outperforms the velocity parameterization on both toy spiral datasets and ImageNet-256. They attribute this behavior to the so-called manifold assumption, under which natural images are assumed to lie on a low-dimensional manifold of the ambient space, making the direct prediction of clean data comparatively simple. In contrast, the velocity and noise parameterizations require the network to predict a term containing noise, thus of full dimension. As a consequence, and as demonstrated numerically by the authors, strongly increasing the dimension degrades the performance of $\mathcal{C}_{\text{vel}}$ and $\mathcal{C}_{\text{noise}}$, while the performance of $\mathcal{C}_{\text{den}}$ remains comparatively stable.

In what follows, we aim to bridge this apparent discrepancy and reconcile our empirical findings with those of Li & He (2025). Unlike the choice of the weighting scheme, **we argue that the optimal parameterization cannot be determined in isolation**. Instead, we show that this choice is strongly tied with the data properties and the chosen architecture.

There are two key differences between the setting of Li & He (2025) and ours (or those of Lipman et al. 2023):

- *Data dimensionality:* So far, our numerics considered resolutions up to 64×64 whereas Li & He (2025) are interested in high-resolution generation, with dimensions up to 1024×1024.
- *Architecture:* Li & He (2025) introduce the JiT architecture, which is essentially a Vision Transformer (ViT, Dosovitskiy et al., 2021) with large patch size, whereas we use U-Nets.

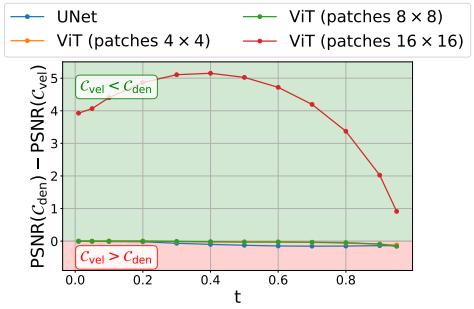
Limited impact of dimensionality alone We first investigate the impact of data dimensionality: we consider CelebA 128×128 while keeping the same U-Net architecture and capacity as for CelebA 64×64: the velocity parameterization still outperforms the denoiser prediction (see Appendix B,Table 5). Regarding generation in pixel space at higher resolutions, Hoogeboom et al. (2023), who consider resolutions up to 512×512, also discard noise prediction due to observed instability at large noise levels and advocate for the velocity parametrization. Overall, this suggests that data dimensionality itself is not the primary factor behind the failure of velocity parametrization observed by Li & He (2025).

The striking impact of architectures: ViTs versus U-Nets We now investigate how changing the network architecture from U-Net to ViT changes the comparison between parametrizations. While U-Nets process images at multiple downscaled resolutions of the initial image using local convolutions, the ViT architecture splits the image into non-overlapping patches that communicate globally through a self-attention mechanism. In Figure 3, we find out that **the patch size crucially affects the performance comparison between velocity and denoiser parametrizations**. For a given data dimension, increasing the patch size reduces the number of input tokens.

- *For small patches*, the velocity parametrization outperforms the denoiser parametrization.
- *For larger patch sizes*, training is faster. As the patch size increases, the overall performance decrease for both parametrizations. Yet, the denoising parametrization demonstrates better robustness, with **complete failure of $\mathcal{C}_{\text{vel}}$ for very large patches**.

Across their experiments, Li & He (2025) keep the ratio between the patch size and the image dimension fixed: regardless of dimension, a patch contains the same amount of information about the image. As a result, the authors report failure of the velocity parametrization in high-resolution settings, where they use large patch size (16 for ImageNet256), while in lower-resolution, with small patch size (4 for ImageNet64), the velocity prediction performs well. In contrast, we show that large patch size alone seems sufficient to explain this phenomenon.

Our experiments reveal a clear link between architectural locality and the preferred training parameterization. Models with limited locality, i.e. ViTs with large patches, perform better when trained to predict the clean data directly. Conversely, models with strong local inductive bias, such as U-Nets and fine-patch ViTs, benefit from the velocity parameterization.



(a) PSNR gap.

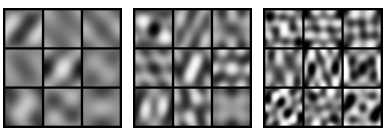
Architecture	$\mathcal{C}_{\text{den}}$	$\mathcal{C}_{\text{vel}}$
ViT/4	19.16	**12.21**
ViT/8	35.05	**30.62**
ViT/16	**79.11**	322.08

(b) Train FID (50k).

Figure 3: Denoising and generation performance of $\mathcal{C}_{\text{vel}}$ versus $\mathcal{C}_{\text{den}}$ when varying the patch size in the ViT architecture. CIFAR-10 (dimension 3×32^2). In the notation ViT/p, p denotes a patch size $p \times p$. Green indicates that $\mathcal{C}_{\text{den}}$ performs better than $\mathcal{C}_{\text{vel}}$, red indicates the opposite.

Impact of the manifold dimension We introduce a synthetic 32×32 grayscale dataset whose samples lie on a controllable, low-dimensional manifold in pixel space. Each image is generated by activating only m selected 2D Fourier modes (with all other frequencies set to zero), sampling the corresponding coefficients, and applying an inverse Fourier transform to obtain a real-valued image (see Figure 4). The integer parameter m therefore directly sets the intrinsic dimension of the data (details in Appendix A).

We study how the intrinsic manifold dimension impacts the relative performance of parameterizations on the synthetic Fourier-32 dataset. For $m \in \{4, 8, 16\}$ we compare $\mathcal{C}_{\text{den}}$ against $\mathcal{C}_{\text{vel}}$. We run the comparison across four architectures: a U-Net, a ViT with 4×4 patches, a ViT with 16×16 patches, and an MLP; all models are tuned to have approximately the same parameter count. Figure 5 reports $\Delta\text{PSNR}(t) := \text{PSNR}(\mathcal{C}_{\text{den}}; t) - \text{PSNR}(\mathcal{C}_{\text{vel}}; t)$ for $m = 4, 8, 16$ and reveals two trends.

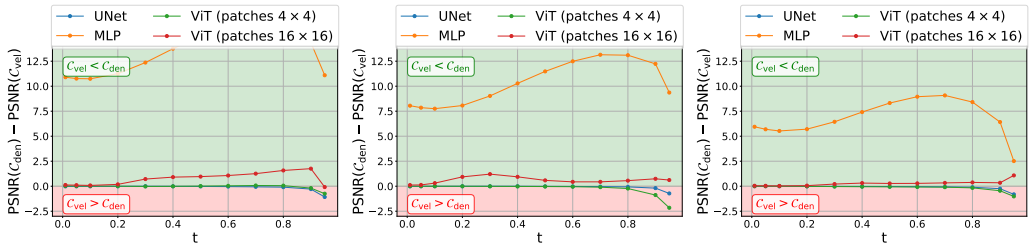


(a) $m = 4$ *(b)* $m = 8$ *(c)* $m = 16$

Figure 4: 9 samples from the Fourier-32 dataset with controlled manifold dimension m.

First, consistently with our CIFAR-10 findings, for a fixed manifold dimension the U-Net and the small-patch ViT favor $\mathcal{C}_{\text{vel}}$ (negative ΔPSNR), whereas the large-patch ViT and the MLP favor $\mathcal{C}_{\text{den}}$.

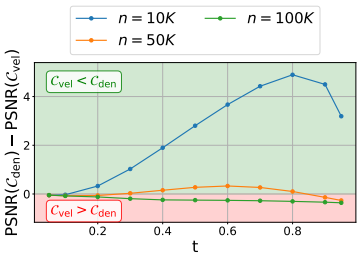
Second, the previously invoked **"manifold assumption" stating that smaller intrinsic dimension should particularly benefit $\mathcal{C}_{\text{den}}$ is only supported for the "coarser models" the ViT-16 and the MLP**: for these models, the advantage of $\mathcal{C}_{\text{den}}$ becomes more pronounced as m decreases. On the other hand, **the U-Net appears insensitive to the manifold dimension** in terms of the $\mathcal{C}_{\text{den}}$ vs. $\mathcal{C}_{\text{vel}}$ ordering. In Appendix A, we check that our PSNR metric correlates with the distance to the manifold.



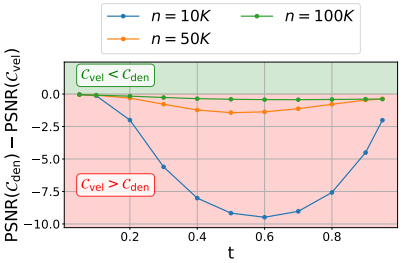
(a) Manifold dim $m = 4$ (b) Manifold dim $m = 8$ (c) Manifold dim $m = 16$

Figure 5: PSNR gap on the Fourier-32 dataset for intrinsic dimensions $m \in \{4, 8, 16\}$, across four architectures. Green indicates that $\mathcal{C}_{\text{den}}$ performs better than $\mathcal{C}_{\text{vel}}$, red indicates the opposite.

Beyond model architectures So far, we have shown that U-Nets are largely unaffected by data dimensionality or manifold dimension, unlike ViTs or MLPs. However, even for U-Nets, other considerations may influence the choice of parametrization. In particular, we show in Figure 6 that the number of available datapoints in the training set is a crucial factor to take into account: in the low data regime, with a fix budget of iterations, the denoiser parametrization largely outperforms the velocity one (Figures 6a and 6c), and, unexpectedly, leads to improved generalization (Figure 6b). Implementation details are given in Appendix A.



(a) PSNR gap, **test data**.



(b) PSNR gap, **train data**.

n	$\mathcal{C}_{\text{den}}$	$\mathcal{C}_{\text{vel}}$
10k	**11.07**	27.25
50k	8.98	**6.89**
100k	9.81	**4.45**

(c) Train FID (50k)

Figure 6: Denoising and generation performance of $\mathcal{C}_{\text{vel}}$ versus $\mathcal{C}_{\text{den}}$ when varying the n, the number of datapoints in the training set. CelebA-64 grayscale (dimension 64^2). Green indicates $\mathcal{C}_{\text{den}}$ performs better than $\mathcal{C}_{\text{vel}}$, red indicates the opposite.

6 Conclusion

Within a unified denoising framework, we compare the weighting losses and parametrizations that naturally arise when training diffusion and flow matching models, namely predicting the clean image, the noise, or the velocity. Isolating each design choice enables us to numerically assess their performance. It shows that **one can benefit from decoupling the natural pairs of weighting and parametrization**, e.g. while noise weighting reaches top performance for all parametrizations, noise parametrization has lowest for all weightings. Regarding the weighting scheme, we confirm that **the weightings** $w^t \propto (1-t)^{-2}$ **are the go-to choice**: we provide a **theoretical justification** based on our denoising formulation. Choosing the right parametrization proves to be more complex: we highlight the role of architectural inductive biases and data properties. Interestingly, our results indicate that **the choice of parameterization should be guided by the degree of locality encoded in the architecture**.

ACKNOWLEDGMENTS

This project was supported by the SHARP project of the PEPR-IA (ANR-23-PEIA-0008, granted by France 2030). Ségolène Martin's work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – The Berlin Mathematics Research Center MATH+ (EXC-2046/1, project ID: 390685689). The authors thank the Blaise Pascal Center for the computational means. It uses the SIDUS (Quemener & Corvellec, 2013) solution.

REFERENCES

Alexander C. Aitken. IV. on least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55:42–48, 1936.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *ICLR*, 2023.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *CVPR*, 2022.

Valentin De Bortoli. Convergence of denoising diffusion models under the manifold hypothesis. *TMLR*, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Ruiqi Gao, Emiel Hoogeboom, Jonathan Heek, Valentin De Bortoli, Kevin Patrick Murphy, and Tim Salimans. Diffusion models and Gaussian flow matching: Two sides of the same coin. In *Blogpost Track at ICLR 2025*, 2025.

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.

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. *NeurIPS*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pp. 13213–13232. PMLR, 2023.

Qing Jin and Chaoyang Wang. Revisiting diffusion model predictions through dimensionality. *arXiv preprint arXiv:2601.21419*, 2026.

Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *NeurIPS*, 2023.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, Toronto, ON, Canada, 2009.

Dibyanshu Kumar, Philipp Vaeth, and Magda Gregorová. Loss functions in diffusion models: A comparative study. In *ECML*, 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

Emmanuel Quemener and Marianne Corvellec. Sidus—the solution for extreme deduplication of an operating system. *Linux J.*, 2013(235), November 2013. ISSN 1075-3583.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.

Anton Schick. Efficient estimates in linear and nonlinear regression with heteroscedastic errors. *Journal of Statistical Planning and Inference*, 58(2):371–387, 1997.

Cosma Rohilla Shalizi. Moving beyond conditional expectations: Weighted least squares, heteroskedasticity, local polynomial regression. Lecture Notes Chapter 7, Department of Statistics, Carnegie Mellon University, 2013. URL https://www.stat.cmu.edu/~cshalizi/uADA/16/lectures/08.pdf.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *TMLR*, 2024.

Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. From facial parts responses to face detection: A deep learning approach. In *ICCV*, 2015.

Hu Yu, Li Shen, Jie Huang, Hongsheng Li, and Feng Zhao. Unmasking bias in diffusion model training. In *ECCV*, 2024.

A IMPLEMENTATION DETAILS

Training details All networks are trained with the same random initialization to ensure comparability. For CIFAR-10 we train for 1000 epochs with batch size 128, and for CelebA-64 we train for 300 epochs with batch size 128. We apply exponential moving average to stabilize training. For CIFAR-10, the UNet architecture is taken from the `torchcfm` library (Tong et al., 2024). For CelebA-64, we use the U-Net architecture (Ronneberger et al., 2015) as in Ho et al. (2020).

The Fourier Dataset Let $N \in \mathbb{N}$ and consider images $x \in \mathbb{R}^{N\times N}$. We generate samples by defining a sparse complex Fourier spectrum $\hat{x} \in \mathbb{C}^{N\times N}$ with exactly m active frequency indices $\{(k_i, \ell_i)\}_{i=1}^{m}$. The set of modes is chosen either deterministically as the m lowest-frequency representatives (`lowfreq`, using the periodic radius $r(k,\ell) = \min(k, N-k)^2 + \min(\ell, N-\ell)^2$) or by a seeded random selection, optionally excluding the DC component $(0, 0)$. For each sample, we draw coefficients a_i i.i.d. (Gaussian or uniform, with a scale parameter), populate $\hat{x}$ at the selected indices and their conjugate-symmetric partners to ensure a real spatial signal, and set all remaining Fourier coefficients to zero. We then compute

$$x = \mathrm{Re}\left(\mathrm{ifft2}(\hat{x})\right),$$

(using orthonormal FFT normalization) to obtain the pixel-space image. Finally, we apply a per-sample normalization nonlinearity (typically $\tanh(\alpha x)$) to bound the dynamic range to $[-1, 1]$ for training stability. Since only the m coefficients $\{a_i\}$ vary while the mode set is fixed, the generative process has exactly m degrees of freedom, yielding a dataset with explicitly controlled intrinsic dimension.

Distance to the Fourier manifold. Let $\mathcal{K} \subset \{0, \ldots, N-1\}^2$ denote the set of active Fourier indices defining the m-dimensional Fourier manifold (including the conjugate-symmetric partners needed for real images). Given a generated image $x \in [-1, 1]^{N\times N}$, we compute the orthonormal 2D Fourier transform $\widehat{x} = \mathcal{F}(x)$ and define the spectral residual energy as the energy outside the manifold support,

$$E_{\mathrm{res}}(x) \;=\; \sum_{(k,\ell)\notin\mathcal{K}} \left|\widehat{x}(k,\ell)\right|^2.$$

Lower values indicate that the generated image is closer to the m-mode Fourier manifold.

Influence of dataset size We train the U-Net architecture with weight w^t_{vel} on Celeba-64 grayscale (image dimension 64^2). The total number of iterations is fixed to 150000. FID is computed with the standard Inception backbone, with 3 channels R, G, B equal.

Table 2: Spectral residual energy mean for the FM loss on submanifold dimension $\dim = 16$, across parametrizations.

	$\mathcal{C}_{\mathrm{vel}}$	$\mathcal{C}_{\mathrm{den}}$	$\mathcal{C}_{\mathrm{noise}}$
Unet	9.70×10^{-4}	1.46×10^{-3}	8.98×10^{-3}
ViT-4	2.40×10^{-3}	5.59×10^{-3}	3.04×10^{-2}
ViT-16	3.23	2.03	3.56
MLP	237.75	7.42×10^{-3}	2072.68

B ADDITIONAL RESULTS ON CIFAR10 AND CELEBA-64

Table 3: PSNR and FID for different time-weightings and parametrizations. PSNR computed on 1000 images; FID on 50k train images; CIFAR-10, 1000 epochs.

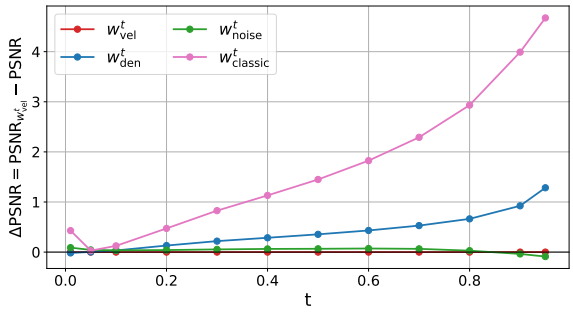
Weighting	**Class**	PSNR (↑)					FID (train 50k) (↓)
		$t = 0.1$	$t = 0.3$	$t = 0.6$	$t = 0.9$	$t = 0.95$	
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{den}}$	14.39	18.20	23.50	32.96	37.41	4.89
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{den}}$	14.39	18.20	23.40	32.64	36.85	16.02
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{den}}$	14.38	18.18	23.52	32.99	37.45	4.60
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{den}}$	14.39	18.03	22.89	30.82	33.18	71.07
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$	14.39	18.21	**23.54**	33.02	37.47	3.90
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{vel}}$	14.39	18.20	23.39	32.67	36.91	10.29
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$	14.37	18.19	23.54	**33.05**	**37.53**	**3.71**
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{vel}}$	14.39	18.03	22.90	31.00	34.46	116.18
$w^t = \frac{1}{(1-t)}$	$\mathcal{C}_{\text{vel}}$	14.42	**18.22**	23.50	32.87	37.30	6.11
$w^t = \frac{1}{(1-t)^3}$	$\mathcal{C}_{\text{vel}}$	14.39	18.15	23.42	32.95	37.47	8.76
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{noise}}$	14.42	18.21	23.50	32.94	37.40	6.30
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{noise}}$	14.41	18.17	23.31	32.35	36.44	20.02
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{noise}}$	14.36	18.20	23.54	33.05	37.53	7.15
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{noise}}$	14.41	18.04	22.89	30.62	34.04	137.55

Table 4: PSNR and FID for different time-weightings and parametrizations. PSNR computed on 1000 test images; FID on 50k train images; CelebA-64, 300 epochs.

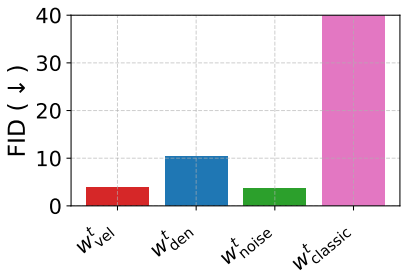
Weighting	**Class**	PSNR (↑)					FID (train 50k) (↓)
		$t = 0.1$	$t = 0.3$	$t = 0.6$	$t = 0.9$	$t = 0.95$	
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{den}}$	15.93	20.79	26.12	34.83	38.81	14.54
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{den}}$	16.01	20.98	26.25	34.50	37.91	19.87
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{den}}$	15.69	20.75	26.26	35.07	39.05	12.11
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{den}}$	15.88	20.19	24.41	29.35	29.81	85.44
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$	**16.03**	**21.08**	**26.52**	35.33	39.34	4.45
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{vel}}$	15.98	20.87	26.10	34.40	38.06	18.83
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$	16.00	21.03	26.46	**35.37**	**39.44**	**3.97**
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{vel}}$	15.89	20.26	24.70	31.34	34.67	133.93
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{noise}}$	15.90	20.85	26.25	35.04	39.05	659.05
$w^t_{\text{den}} = 1$	$\mathcal{C}_{\text{noise}}$	15.91	20.75	25.93	33.64	36.66	95.02
$w^t_{\text{noise}} = \frac{t^2}{(1-t)^2}$	$\mathcal{C}_{\text{noise}}$	15.43	20.97	26.49	35.41	39.49	681.05
$w^t_{\text{classic}} = \frac{1}{t^2}\mathbf{1}_{t>t_{\min}}$	$\mathcal{C}_{\text{noise}}$	15.77	20.34	24.80	30.49	33.87	206.20

Table 5: PSNR and FID for different time-weightings and parametrizations. PSNR computed on 1000 test images; FID on 50k train images; CelebA-128, 300 epochs.

Weighting	**Class**	PSNR (↑)					FID (train 50k) (↓)
		$t = 0.1$	$t = 0.3$	$t = 0.6$	$t = 0.9$	$t = 0.95$	
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{den}}$	17.75	22.71	28.03	36.28	40.17	23.0
$w^t_{\text{vel}} = \frac{1}{(1-t)^2}$	$\mathcal{C}_{\text{vel}}$	18.03	**23.31**	**28.55**	**36.82**	**40.79**	**5.62**

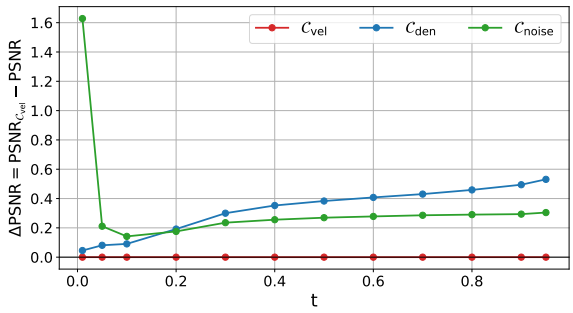


(a) Difference in PSNR (lower is better) between the reference w^t_{vel}, and various models, computed on 1000 test images. The parametrization class is here set to $\mathcal{C}_{\text{vel}}$.

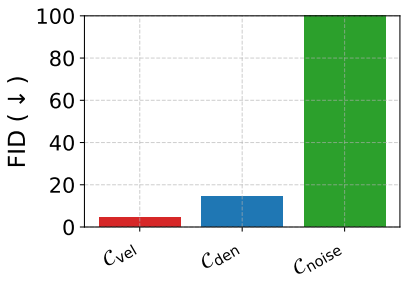


(b) FID on 50k train images (lower is better).

Figure 7: PSNR and FID for the different losses, CelebA-64. Models that reach the highest PSNR (low difference in PSNR compared to standard FM) also reach the lowest FID.



(a) Difference in PSNR (lower is better) between the reference $\mathcal{C}_{\text{vel}}$, and other parametrizations, computed on 1000 test images. The weight is set to w^t_{vel}.



(b) FID on 50k train images (lower is better).

Figure 8: PSNR and FID for the different parametrizations, CelebA-64. Models that reach the highest PSNR (low difference in PSNR compared to standard FM) also reach the lowest FID.