Stable Mean Flow: Lyapunov-Inspired One-Step Flow Matching

Guangxun Zhang Mason Haberle Davi Geiger
New York University
gz2260@nyu.edu mason.haberle@nyu.edu dg1@nyu.edu

Abstract

The Mean Flow Matching algorithm is the state-of-the-art for one-step generative models. Building on this idea, we propose the Stable Mean Flow algorithm and introduce a Lyapunov-inspired stability regularizer that enforces local non-expansivity of the single-step transport map. This design guarantees uniqueness of characteristics and bounds trajectory drift. We conduct experiments that show improved output quality and convergence speed over Mean Flow. Moreover, we establish explicit upper bounds on error growth for both one-step and multi-step generation.

1. Introduction

In the past decade, generative models have achieved remarkable progress in high-fidelity image synthesis [6, 10, 16, 38, 39]. Diffusion and score-based model approaches represent a major breakthrough, but they typically require hundreds or thousands of iterative denoising steps, even when accelerated by few-step samplers or model-distillation techniques [14, 16, 23, 29, 30]. Flow-based generative formulations alleviate this inefficiency by learning deterministic dynamics that transport noise to data in a single pass [25].

Within this family, flow matching treats generation as following a learned velocity field and has inspired work on straightening trajectories to cut solver steps, including rectified-flow and consistency-style techniques [26, 40].

Despite rapid progress, one-step methods can be fragile in practice [9, 17, 31, 41]. Training may exhibit exploding sensitivities, ambiguous trajectories and so on, all of which could lead to instability and bad sample quality. The reason behind these issues is that the learned one-step map can be locally expansive, amplifying perturbations and making trajectories ill-posed because of lack of ground-truth target[9].

We introduce Stable Mean Flow, a Lyapunov-inspired framework for one-step flow matching that encourages local non-expansivity of the update map while preserving the mean-flow objective [9]. Inspired from the idea of Lyapunov stability in dynamical systems, we add a simple hinge-style stability loss that discourages local stretching of nearby points. Under this stability-aware objective, we establish uniqueness of trajectories for the learned dynamics and derive practical bounds on sensitivity that curb numerical pathologies often seen during one-step training. We further develop forward one-step error recursions and near-terminal endpoint error control.

Empirically, Stable Mean Flow not only improves one-step Fréchet Inception Distance on CIFAR-10 relative to Mean Flow while preserving single-call sampling, but also exhibits markedly faster convergence during training. Both mid-training qualitative comparisons and our checkerboard analysis show that Stable Mean Flow aligns with the target geometry significantly earlier, forming sharper and more coherent structures in the early stage. Ablations further reveal a narrow yet reliable hyperparameter regime for the perturbation radius and stability weight. Taken together, these results suggest that stability-aware regularization provides a minimal, theoretically grounded, and practically effective pathway toward faster and more reliable one-step generation, complementing ongoing advances in straightened flow matching [9, 21, 25, 26, 40].

2. Related Work

2.1. History of Generative Models

Over the past decade, generative modeling has progressed at remarkable speed. Diffusion models [16], introduced by Ho et al., generate data by reversing a gradual noising process, while score-based methods [23, 32] learn the score function directly and enable reverse-time SDE sampling, thereby unifying diffusion and score matching. Despite their strong empirical performance, both frameworks typically require hundreds or even thousands of denoising steps, leading to slow sampling.

Few-step diffusion techniques reduce sampling cost by shortening denoising trajectories—e.g., DDIM and distillation [28–30]—but they still rely on iterative refinement. A more fundamental shift arises in flow-based generative modeling: flow matching [1, 5, 12, 20, 25, 34, 35] learns continuous-time ODEs that transport noise to data, connect-

ing diffusion and normalizing flows under optimal-transport viewpoints [18]. Consistency Models [8, 33] similarly enable few-step generation via multi-scale self-consistency, and recent analysis shows they are closely related to flow-map matching and stochastic interpolants [4].

To further reduce solver steps, straight-trajectory flow matching seeks nearly linear generative paths [24, 27]. Rectified Flow progressively "reflows" trajectories toward straight endpoints [13, 26], while Consistency Flow Matching shapes straight paths via segment-wise velocity consistency [40]. At the extreme, one-step flow matching collapses the entire trajectory into a single map: Optimal Flow Matching uses convex-potential gradients to approximate quadratic-cost OT displacements [21], whereas Mean Flow [9] introduces an analytically defined average-velocity target enabling stable from-scratch training and true 1-NFE generation.

2.2. Flow Matching

Flow Matching is a deterministic, ODE-based approach of generative modeling that learns how to continuously transport a simple prior (e.g., Gaussian noise) into the data distribution [25]. Instead of learning probabilities directly, FM trains a neural *velocity field* that tells each point how to move over time so that a cloud of noise transforms into realistic data [2, 25]. A convenient way to describe intermediate states is a simple interpolation between data x and noise ϵ:

$$z_t = a_t\, x + b_t\, \epsilon,$$

where (a_t, b_t) is a fixed schedule. The model is trained to predict the direction in which z_t should move. Because many (x, ϵ) pairs can yield the same z_t, FM uses a *conditional* training objective that is easy to sample from and is provably equivalent (in expectation) to matching the true underlying flow [25]:

$$L_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x,\epsilon} \| \, v_\theta(z_t, t) \; - \; v_t(z_t \mid x) \, \|^2 .$$

Here v_θ is the learnable velocity field and $v_t(\cdot \mid x)$ is the per-sample target direction induced by the chosen path. After training, generation amounts to following the learned dynamics from noise to data by solving the ODE

$$\frac{d}{dt} z_t = v_\theta(z_t, t) \qquad \text{with} \quad z_1 \sim p_{\text{prior}}.$$

This framework subsumes diffusion-style paths but also supports more efficient choices like Optimal Transport (OT) paths [2], which tend to be straighter and enable faster sampling while retaining stability and quality. Related straightening ideas include Rectified Flow, which iteratively "rewires" paths toward straight trajectories to reduce solver steps [26].

2.3. Mean Flow Matching

Mean Flow revisits what the model should learn. Instead of the *instantaneous* velocity used in FM (the tangent at time t), Mean Flow trains a network to predict the *average* velocity—the displacement per unit time across a time interval [9]. Formally, if v denotes the instantaneous field, the average field between r and t is

$$u(z_t, r, t) = \frac{1}{t-r} \int_r^t v(z_\tau, \tau)\, d\tau.$$

In practice, the network learns u_θ so that a complete sample can be produced in a *single* step via the displacement

$$z_r \approx z_t - (t-r)\, u_\theta(z_t, r, t) \qquad \text{with } (r,t) = (0,1)).$$

Because the target is a well-defined physical quantity (average velocity) rather than a heuristic constraint, training is relatively stable [9]. Here $t \geq r$ are two time coordinates on $[0,1]$ that index the averaging window; t is the current time and r is the reference time. During training, (t, r) are sampled; for one-step sampling we use $(r, t) = (0, 1)$.

Let z_t be the interpolation path (e.g., $z_t = (1-t)x + t\varepsilon$), and define the instantaneous velocity at time t by $v_t := \frac{dz_t}{dt}$ (for the standard FM path, $v_t = \varepsilon - x$). The Mean Flow loss is

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\big\| \, u_\theta(z_t, r, t) - \mathrm{sg}(v_t - (t-r)\, \frac{d}{dt} u_\theta(z_t, r, t)) \big\|_2^2 \Big]$$

The total derivative is computed as a Jacobian–vector product (JVP):

$$\begin{aligned} \frac{d}{dt} u_\theta(z_t, r, t) &= \partial_z u_\theta(z_t, r, t)\, v_t + \partial_t u_\theta(z_t, r, t) \\ &\equiv \mathrm{JVP}\big(u_\theta;\, (z_t, r, t)\big). \end{aligned}$$

One-step sampling with $(r, t) = (0, 1)$ uses

$$z_0 \approx z_1 - u_\theta(z_1, 0, 1) \quad \text{(e.g., } x \approx \varepsilon - u_\theta(\varepsilon, 0, 1)).$$

3. Stable Mean Flow Matching

For current flow matching models, such as Consistency Flow [40] and Rectified Flow [26], the focus is on the straight-line trajectory defined by

$$x(t) = tX_1 + (1-t)X_0.$$

One motivation for this choice comes from Brenier's theorem in optimal transport [36], which states that under a quadratic cost, optimal transport plans follow a straight-line interpolation between source and target almost everywhere. In this setting, the geodesic is exactly the linear path above.

Taking the derivative of both sides yields a simple ordinary differential equation:

$$\dot{x}(t) = X_1 - X_0.$$

Building on this ODE, the central idea of the Stable Mean Flow Matching algorithm we propose is to combine the trajectory guidance of Mean Flow with Lyapunov stability theory in order to improve both training dynamics and generation quality. Intuitively, Mean Flow Matching acts like a GPS that tells you where to go, while the new Lyapunov stability term acts as traction control, ensuring the wheels stay firmly on the planned road and preventing deviations from the best trajectory.

3.1. Lyapunov Stability for the Constant ODE

Below is the classical definition of stability in dynamical systems [3].

Definition 1 (Lyapunov Stability)**.** *Let $x(t; t_0, x_0)$ denote the solution of an ODE with initial condition $x(t_0) = x_0$, defined for all $t \geq t_0$. A reference solution $x^*(t) := x(t; t_0, \bar{x}_0)$ is* Lyapunov stable *if, for every $\varepsilon > 0$, there exists $\delta > 0$ such that*

$$\|x_0 - \bar{x}_0\| < \delta \implies \|x(t; t_0, x_0) - x^*(t)\| < \varepsilon \quad \forall\, t \geq t_0.$$

We consider the conditional target dynamics

$$\dot{x}(t) = a, \qquad a := X_1 - X_0 \text{ (constant)}.$$

Intuitively, every point in space receives the *same* push at every moment. Two trajectories therefore drift in parallel, never squeezing together or blowing apart. Their separation is frozen in time:

$$\|x(t) - y(t)\|_2 = \|x_0 - y_0\|_2 \quad \text{for all } t \geq 0.$$

This immediately gives Lyapunov stability of any reference solution $x^*(t)$: if we start close, we stay exactly that close forever. Formally, for any tolerance $\varepsilon > 0$, choose the initial tolerance $\delta = \varepsilon$; then

$$\|x_0 - \bar{x}_0\|_2 < \delta \implies \|x(t) - x^*(t)\|_2 < \varepsilon \quad \text{for all } t \geq 0.$$

Geometrically, the reference trajectory carves out a straight path, and every nearby trajectory lives inside a time-invariant tube whose radius equals the initial gap. This "parallel drift" picture is precisely the trapping behavior we seek when encouraging stability in learned flows as shown in Figure 1.

3.2. Consequences of the Stability Constraint

For $r \in [0, 1)$ and $t \in (r, 1]$, we formalize a one–step map that moves a state z at time t along the learned velocity field u_θ with a backward-in-time Euler–style update:

$$\phi_r^\theta(t, \cdot): \quad z \longmapsto z - (t - r)\, u_\theta(z, r, t).$$

Choice of Objective Function Based solely on the Lyapunov stability definition (see Def. 1), a natural local constraint at a base point z_t is:

$$\left\|\phi_r^\theta(t, z_t + \Delta z) - \phi_r^\theta(t, z_t)\right\|_2 \;\leq\; \delta, \qquad \forall\, \Delta z:\ \|\Delta z\|_2 \leq \delta. \qquad (\delta\text{-cap})$$

However, in practice, we replace (δ-cap) with the *non-expansivity* (NE) condition (NE):

$$\left\|\phi_r^\theta(t, z_t + \Delta z) - \phi_r^\theta(t, z_t)\right\|_2 \;\leq\; \|\Delta z\|_2\,, \qquad \forall\, \Delta z:\ \|\Delta z\|_2 \leq \delta. \qquad (\text{NE})$$

The NE constraint is stronger and cleaner. It automatically implies δ-cap for any chosen δ, while also ensuring that infinitesimal perturbations are not amplified. In contrast, δ-cap alone can still allow sharp local stretching as long as the displacement stays within the fixed radius δ.

Uniqueness of trajectories implied by NE Uniqueness is crucial in flow matching. The training objective regresses a forces vector field u_θ to a target field along sample paths. When the dynamical system generates unique characteristics, each initial state is transported along a single, well-defined trajectory and there is no ambiguity about which path a sample follows. In Optimal Transport terms, the pushforward measures induced by the learned flow are unambiguous.

Theorem 3.1 (Uniqueness of characteristics under NE)**.** *Suppose the learned field $u_\theta(\cdot, r, \cdot)$ follows the Stable Mean Flow vector field Then, for every initial condition z_0 and t_0, the ODE*

$$\dot{z}(s) = u_\theta\big(z(s), r, s\big), \qquad z(t_0) = z_0$$

admits a unique trajectory $z(s)$.

This prevents crossing or multi-valued transports that would otherwise make the regression target inconsistent across time steps. Uniqueness also stabilizes gradient propagation and aligns the learned map with a deterministic coupling between source and target distributions, which improves sample quality and training stability in conditional and unconditional flow matching alike.

Controllable JVP Stability under NE A major practical failure case of Mean Flow matching is that the Jacobian–vector product (JVP) sometimes produces `NaN`/`Inf` and ruins training [9]. Local non-expansivity (NE) of the update map $\phi_r^\theta(t, \cdot) = \mathrm{Id} - \alpha\, u_\theta(\cdot, r, t)$ with $\alpha := t - r > 0$ yields theoretical and explicit *numerical* control of the JVP magnitude.

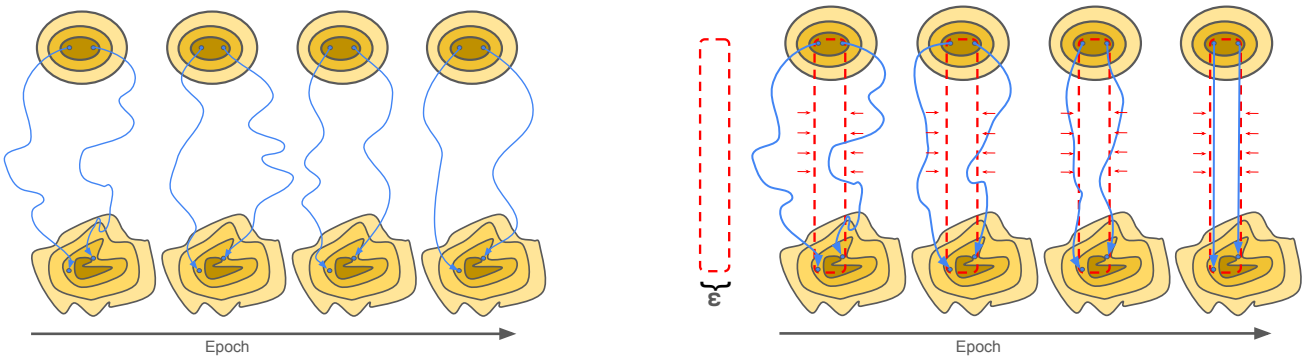


Figure 1. The left figure illustrates trajectories of Mean Flow; the right figure illustrates trajectories of Stable Mean Flow.

Theorem 3.2 (Bounded JVP)**.** *Let $\phi_r^\theta(t,z)$ be defined as above. Then, in the Stable Mean Flow algorithm, the JVP is bounded by a constant C:*

$$\|\partial_z u_\theta\, \xi + \partial_t u_\theta\| \;\le\; C.$$

This control removes one of the most common numerical failure modes in Mean Flow Matching. The update operator cannot amplify perturbations without limit, avoiding uncontrolled error blow-up. Importantly, this regularization does not bias the learning of the JVP when the ground-truth solution already satisfies local non-expansivity; it merely keeps the optimization trajectory inside a numerically safe region. As shown in Figure 2, Stable Mean Flow maintains relatively more stable JVP behavior during training.

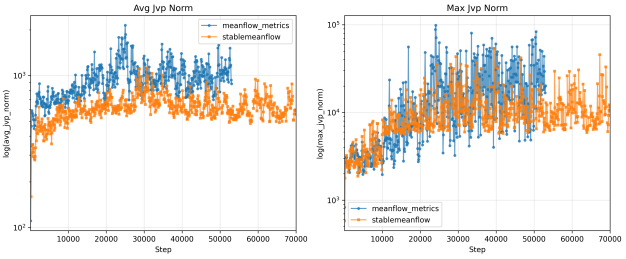


Figure 2. Comparison of average and maximum JVP values in CIFAR-10 experiment. Mean Flow collapses around 53,000 iterations.

Choice of δ One of the most important hyperparameters in Stable Mean Flow is the radius δ: perturbations are random by drawing a Gaussian vector $\xi \sim \mathcal{N}(0, I_d)$, normalizing to the unit sphere, and then scaling to radius δ:

$$\Delta z = \delta\, \frac{\xi}{\|\xi\|_2}, \qquad \xi \sim \mathcal{N}(0, I_d).$$

The choice of large or small δ affects the goal of training. We first analyze local behavior at a fixed base point z_t. All expectations and probabilities are taken with respect to Δz while conditioning on z_t.

Theorem 3.3 (Small-Radius Robustness Bound)**.** *Let $\alpha = t - r > 0$. Define the signed violation*

$$V(\Delta z) := \left\|\phi_r^\theta(t, z_t + \Delta z) - \phi_r^\theta(t, z_t)\right\|_2 - \|\Delta z\|_2.$$

Then, for $\Delta z \sim \mathcal{D}_\delta$ (conditional on z_t),

$$\mathbb{E}\big[V(\Delta z)\big] \;\le\; 2\,\delta, \qquad \mathbb{P}(V(\Delta z) > \tau) \;\le\; \frac{2\,\delta}{\tau} \quad (\tau > 0).$$

As δ increases, the upper bound of $\mathbb{E}\big[V(\Delta z)\big]$ also increases. In practice, too large a δ makes our stochastic estimate for the objective unreliable, hindering regression to the Mean Flow target. Too small a δ yields vanishing gradients, returning to standard Mean Flow. We therefore use a *small but finite* radius (e.g. a fixed fraction of the typical step scale $\alpha\|u_\theta\|$ or of the local state norm), ensuring effective stabilization without degrading learning.

Local non–expansivity and soft penalty. During the training, we impose *non-expansivity* (NE) softly via the hinge–squared penalty

$$\ell_{\text{stab}}(\Delta z) = \Big[\max\big\{0,\; \|\Delta z - (t-r)\,\Delta u\|_2 - \|\Delta z\|_2\big\}\Big]^2,$$

where $\Delta u := u_\theta(z_t + \Delta z, r, t) - u_\theta(z_t, r, t)$ and $\alpha := t - r > 0$. The loss ℓ_{stab} is zero whenever (NE) holds and increases quadratically otherwise. Our final objective is a weighted combination of the Mean Flow loss and the stability loss, $L = \mathcal{L} + \mu\ell_{\text{stab}}$. By choosing proper hyperparameters, Stable Mean Flow could achieve coherent image generation. The SMFM algorithm is shown in Algorithm 1.

Algorithm 1 Hybrid Mean Flow with Non-Expansivity

Require: Initial state θ_0, learning rate $\eta > 0$, stability weight $\mu > 0$, perturbation size $\delta > 0$, iterations T

Initialize $\theta \leftarrow \theta_0$
for $i = 1$ to T **do**
  Sample $t \sim \mathrm{Unif}[\varepsilon, 1]$, $\quad r \sim \mathrm{Unif}[0, t]$
  Sample $z_t \sim p_t$, $\quad \xi \sim \mathrm{Unif}(\mathbb{S}^{d-1})$, $\quad \Delta z \leftarrow \delta\,\xi$
  $u \leftarrow u_\theta(z_t, r, t)$, $\quad v \leftarrow v(z_t, t)$
  $u_{\mathrm{tgt}} \leftarrow v - (t-r)\left(\partial_z u_\theta(z_t, r, t)\, v + \partial_t u_\theta(z_t, r, t)\right)$
  $L_{\mathrm{MF}} \leftarrow \|u - \mathrm{sg}(u_{\mathrm{tgt}})\|_2^2$
  $\Delta u \leftarrow u_\theta(z_t + \Delta z, r, t) - u$
  $L_{\mathrm{stab}} \leftarrow \max\!\big(0,\ \|\Delta z - (t-r)\Delta u\|_2 - \|\Delta z\|_2\big)^2$
  $L \leftarrow L_{\mathrm{MF}} + \mu\, L_{\mathrm{stab}}$
  $\theta \leftarrow \theta - \eta\, \nabla_\theta L$

4. Endpoint Control

After training the final model u_θ, we examine its temporal behavior in order to translate the terminal data-facing quantity (the endpoint velocity error at $s{=}1$) into guarantees about the *entire backward trajectory*. We especially care about how velocity errors evolve over time. A minimal set of regularity conditions (i) make the flow well-posed, (ii) control its sensitivity to perturbations, and (iii) ensure all derived bounds remain *uniform* across the relevant state space and parameter set:

Standing assumptions. Fix $r \in [0, 1)$ and compact sets $K \subset \mathbb{R}^d$ and Θ. For each $\theta \in \Theta$:

(S1) $u_\theta(\cdot, r, \cdot) \in C^1(K \times [r, 1])$.
(S2) u^* is continuous on $K \times [r, 1]$.
(S3) The segment $s \mapsto z_s$ remains in K for $s \in [r, 1]$.

Immediate bounds. By compactness and continuity, there exist finite constants

$$M_\theta := \max \|u_\theta\| < \infty, \qquad M^* := \max \|u^*\| < \infty.$$

Moreover, letting

$$\Lambda_\theta := \max \|\partial_t u_\theta\| < \infty,$$

the mean–value theorem in t gives the uniform Lipschitz-in-time bound

$$\|u_\theta(t) - u_\theta(t-\tau)\| \ \le\ \Lambda_\theta\, |\tau|, \qquad t, t-\tau \in [r, 1].$$

Endpoint error and its role.

Endpoint error and its role. Let $u^*(s, z)$ denote the *oracle velocity*, defined as the conditional expectation of the path tangent at time s given the state $z_s = z$:

$$u^*(s, z) := \mathbb{E}[\dot{z}_s \mid z_s = z], \qquad \dot{z}_s := \frac{d}{ds} z_s,$$

where the expectation is taken over the randomness used to generate the path

For the straight-line interpolation used in our one-step setting, the path velocity is constant, so the endpoint satisfies the oracle identity

$$z_r \ =\ z_1 - (1-r)\, u^*(1, z_1).$$

We define the *terminal velocity error* at $s = 1$ by

$$e_1 \ :=\ u_\theta(z_1, r, 1) - u^*(1, z_1).$$

Since the one-step sampler uses the learned velocity u_θ at $(z_1, 1)$, its reconstruction is

$$\widehat{z}_r \ =\ z_1 - (1-r)\, u_\theta(z_1, r, 1),$$

which gives

$$\widehat{z}_r - z_r \ =\ -(1-r)\, e_1.$$

Endpoint accuracy is controlled *directly* by the terminal velocity mismatch: making e_1 small linearly shrinks the sampling error at time r. Early-time errors e_r do not enter this one–step relation, so a model can still produce an accurate $\widehat{z}_r$ provided the terminal field is correct. This is why reducing the endpoint error near $s = 1$ is central for sample quality and motivates our focus on controlling the terminal velocity.

Theorem 4.1 (Forward one-step error bound)**.** *Fix $t \in [r, 1)$ and $\Delta t \in (0, 1-t]$ and set $\alpha_t := t - r$. Let $\{z_s\}_{s\in[r,1]}$ satisfy the oracle identity defined above. Then, we have*

$$\|e_{t+\Delta t}\| \ \le\ \frac{\alpha_t}{\alpha_t + \Delta t}\, \|e_t\| \ +\ \frac{\|z_{t+\Delta t} - z_t\|}{\alpha_t + \Delta t} + \frac{\Delta t\big(M_\theta + \alpha_t \Lambda_\theta\big)}{\alpha_t + \Delta t}.$$

The forward one–step error bound in **Theorem 4.1** provides a quantitative inequality that relates the error at time $t + \Delta t$ to the error at time t. While this controls each update, it does not yet describe the multi-step error dynamics: can large step sizes cause blow-up, is stability preserved below a threshold, and do errors above the threshold decrease? These properties are crucial for ensuring that the training dynamics remain well–behaved without requiring excessively small time steps.

The following **Corollary 4.1** addresses these questions by making the one–step inequality global.

Corollary 4.1 (Non-Growing Upper Bound)**.** *Fix $r \in [0, 1)$ and a time grid $r \le t_0 < t_1 < \cdots \le 1$ with $\Delta t_k := t_{k+1} - t_k > 0$ and $\alpha_{t_k} := t_k - r$. Define*

$$T_{t_k} \ :=\ M^* + M_\theta + \alpha_{t_k} \Lambda_\theta.$$

Then,

$$\|e_{t_{k+1}}\| \ \le\ \max\{\|e_{t_k}\|,\ T_{t_k}\}.$$

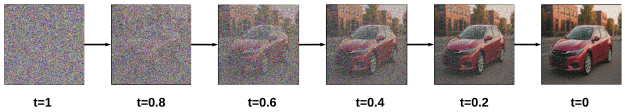


Figure 3. Stable Mean Flow converges backward in time from $s = 1$ to $s = 0$ under multi-step generation

This mechanism creates a safety envelope that prevents blow-ups, confines the error to a predictable region, and triggers contraction once the deviation exceeds a threshold. Figure 3 illustrates the backward convergence of Stable Mean Flow under multi-step generation.

A defining property of the Stable Mean Flow model is that sampling depends only on the velocity field at the terminal time $t = 1$. Consequently, the quality of a generated sample is directly governed by the endpoint instantaneous error $\|e_1\|$. The behavior of the flow at intermediate times influences the sampler only through its effect on this final quantity. Corollary 4.2 provides an explicit upper bound on $\|e_1\|$ in terms of the error at any earlier time $t < 1$ together with the stability constants of the system.

Corollary 4.2 (Endpoint control)**.** *Let* $e_s := u_\theta(z_s, r, s) - u^*(s, z_s)$ *and* $\alpha_s := s - r$. *For any* $t \in [r, 1)$,

$$\|e_1\| \;\le\; \frac{\alpha_t}{\alpha_1}\,\|e_t\| \;+\; \frac{1-t}{\alpha_1}\,(M^* + M_\theta) \;+\; \frac{\alpha_1^2 - \alpha_t^2}{2\alpha_1}\,\Lambda_\theta, \qquad \alpha_1 = 1 - r.$$

For one-step generation, this gives the upper bound

$$\|e_1\| \;\le\; (M^* + M_\theta) \;+\; \tfrac{1}{2}\,\Lambda_\theta.$$

This corollary furnishes a theoretical upper bound for both one-step generation $e_0 \to e_1$ and multi-step schedules $e_0 \to e_{t_1} \to \cdots \to e_{t_n}$ with $r = t_0 < t_1 < \cdots < t_n = 1$. If the model's velocity field u_θ agrees with the oracle u^* on a small window near $t = 1$, the endpoint z_1 remains close to the oracle trajectory; the residual error can increase at most linearly in the remaining time-to-go. This provides a theoretical guarantee of endpoint stability.

5. Experiments and Comparisons

Experimental setup. We evaluate *Stable Mean Flow Matching* (SMFM) on CIFAR-10 to assess stability, sample quality, and efficiency under single-step inference [22]. We use Fréchet Inception Distance (FID; lower is better) as the main metric [15]. All runs in Sec. 5.1 are trained for 500k iterations with batch size 128 on a single NVIDIA A100.

For the runs in Sec. 5.2, SMFM is trained under an identical training process to Mean Flow [9] and is evaluated with the same single-step sampler (NFE = 1). We compute FID against the CIFAR-10 test statistics following common practice.[1]

Model and training details. In the experiments below, the stability term is activated in a early time window and then its weight μ decreases as iterations grow in order to avoid overwhelming the Mean Flow objective. Hyperparameters are selected during ablation in Sec. 5.1. Data augmentations and optimization schedules mirror those used for Mean Flow [9]; thus any performance differences can be attributed to the stability mechanism rather than incidental training choices.

5.1. Hyperparameter Study

We conduct a sweep of the two important hyperparameters for SMFM with less iterations and batch sizes: the perturbation radius Δz and the stability weight μ. Resulting FID on CIFAR-10 are depicted in Table 1 and Figure 4. The model was trained for 500k iterations with batch size 128 on a single NVIDIA A100.

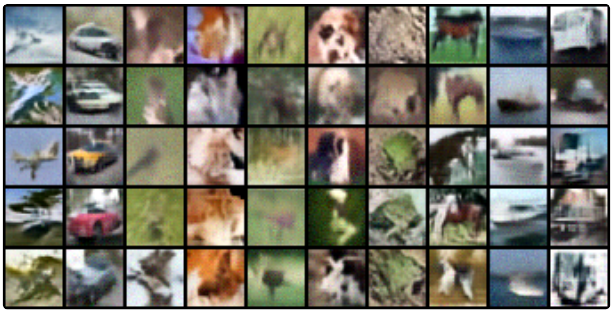
Table 1. **CIFAR-10 FID** over a grid of $(\Delta z, \mu)$; lower is better. Bold indicates the best observed setting within these sweeps.

$\Delta z \backslash \mu$	0	0.1	0.5	1
0.005	–	85.41	127.40	224.53
0.01	86.73	**79.86**	134.53	253.67
0.02	–	95.17	187.43	331.23

Figure 4. Bubble chart comparing one-step FID varying Δz and μ

Findings and interpretation. SMFM exhibits a pronounced sensitivity to both Δz and μ. Small Δz yields a stability signal that is informative and localized, while a

[1] The precise evaluation budget (e.g., number of generated images) follows standard CIFAR-10 settings used by prior Mean Flow Matching



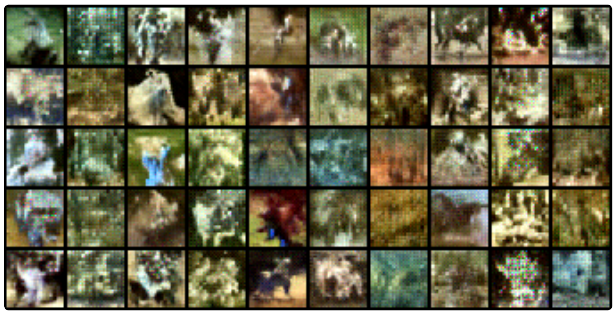


Figure 5. Qualitative comparison at 200k iterations (batch size 128, NFE = 1). *Left:* Stable Mean Flow (SMFM). *Right:* Mean Flow.

modest μ balances this signal against the mean-flow objective. In contrast, larger values of either quickly degrade sample quality: the regularizer dominates, over-contracts the dynamics, and biases the learned field away from the mean-flow target. Practically, we recommend keeping Δz small and scale-aware and adjusting μ gently; the effective regime is a narrow band where the constraint improves conditioning without overwhelming the base objective.

Table 2. **CIFAR-10 comparison.** 1- and 2-step generative models. NFE: number of function evaluations; Lower FID is better.

Method	NFE	FID
1-Rectified Flow [26]	1	378
Glow [19]	1	48.9
Residual Flow [7]	1	46.4
GLFlow [37]	1	44.6
DenseFlow [11]	1	34.9
Consistency Model [31]	2	5.83
Consistency Flow Matching [40]	2	5.34
Mean Flow [9]	1	2.92
Stable Mean Flow (ours)	1	**2.86**

5.2. CIFAR-10 comparisons

Table 2 situates SMFM among representative one-step and few-step methods under matched single-step inference. SMFM improves FID from 2.92 to 2.86 over Mean Flow while preserving 1-step sampling efficiency, indicating that the stability term enhances robustness without sacrificing inference speed after training.

Convergence and qualitative results. Figure 5 illustrates representative generations at a matched mid-training checkpoint (200k iterations, batch size 128, NFE = 1). SMFM converges faster to visually coherent samples than Mean Flow, consistent with its lower FID at NFE = 1. However, as NFE becomes 5, there is no huge difference both visually and in FID.

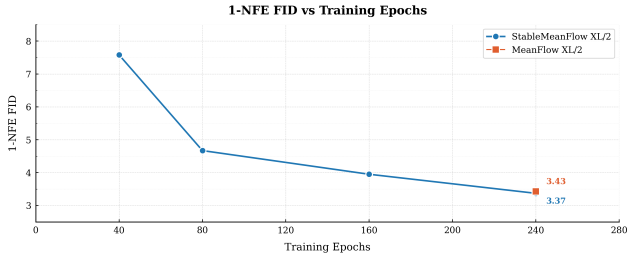


Figure 6. Comparison of SMF-XL/2 with MeanFlow-XL/2 on ImageNet

5.3. ImageNet comparisons

We further validate our results with new experiments. With the same training process, we trained a single Stable MeanFlow-XL/2 model to compare our results with those reported in the original MeanFlow paper. In Figure 6, by Epoch 240, Stable Mean Flow achieves an FID of 3.37. This is slightly lower than the 3.43 reported for MeanFlow-XL/2, though the difference is not substantial. Based on Figure 4 in the original MeanFlow paper [9], Stable MeanFlow exhibits noticeably better performance early on in training.

5.4. Early-Stage Convergence in a Toy Example

In order to better visulize the trajectory of Stable Mean Flow, we employ the 2D checkerboard as a controlled testbed to investigate *early-stage convergence* and *one-step* generation quality under different flow-learning objectives. Our study focuses on comparing the baseline Mean Flow model with the proposed Stable Mean Flow, both trained under identical settings for 100,000 iterations with a batch size of 1024 on NVIDIA GeForce RTX 5090 GPU.

Figure 7 illustrates the qualitative evolution of the learned density fields for both **Stable Mean Flow** (top) and the original **Mean Flow** (bottom) under identical train-

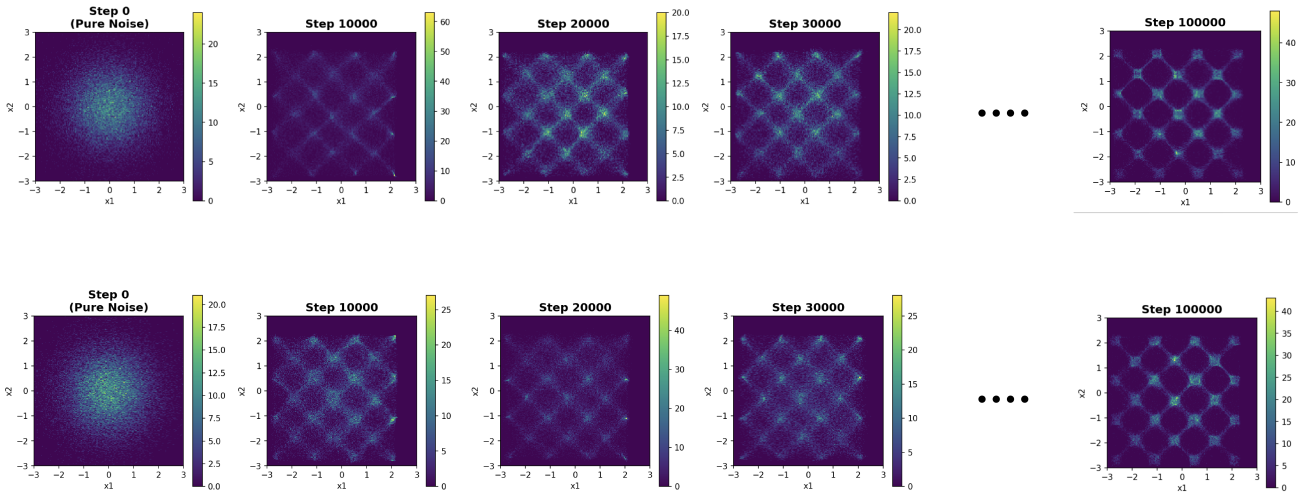


Figure 7. Training dynamics on the checkerboard distribution with one-step sampling. **Top:** Stable Mean Flow. **Bottom:** Mean Flow.

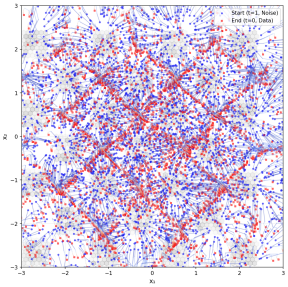


Figure 8. Particle trajectories from noise to learned checkerboard distribution for Stable Mean Flow with 5000 particles in the velocity field in Step 10000

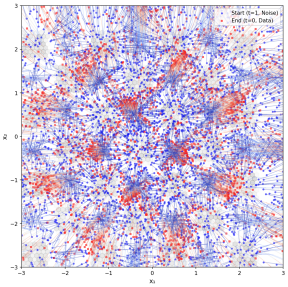


Figure 9. Particle trajectories from noise to learned checkerboard distribution for Mean Flow with 5000 particles with velocity field in in Step 10000

ing settings. In Figure 8 and Figure 9, Stable Mean Flow already concentrates the majority of probability mass on the correct lattice sites, forming distinct, high-contrast modes with minimal background noise, while the original Mean Flow remains more diffuse and less geometrically aligned. As training continues, the baseline model gradually closes the gap, but Stable Mean Flow consistently maintains sharper boundaries and cleaner separations between neighboring cells.

In the early stage, the stability constraint compels the velocity field to first learn correct vector orientations before increasing magnitudes, enabling trajectories to converge smoothly into their respective basins instead of oscillating across cell boundaries. Consequently, particle motion becomes more structured and monotonic, and the target distribution is visually organized faster.

6. Conclusion and Future Work

Stable Mean Flow adds a lightweight, Lyapunov-inspired non-expansivity prior to one-step flows, producing faster early alignment and cleaner geometry at 1-NFE. We see consistent early- and mid-training gains, with more stable characteristics and less overshoot. There is a dramatic early boost in convergence compared to mean flow. The method also provides upper bounds on the error growth for both one-step and multi-step generation. In future work, we aim to further improve the gains we see in endpoint performance and optimize our algorithm to better narrows the training time.

References

[1] Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. 1

[2] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022. 2

[3] Luis Barreira and Claudia Valls. *Ordinary differential equations: Qualitative theory*. American Mathematical Soc., 2012. 3

[4] Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*, 2025. 2

[5] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024. 1

[6] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 1

[7] Ricky T. Q. Chen, Jens Behrmann, David K Duvenaud, and Joern-Henrik Jacobsen. Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 7

[8] Zhengyang Geng, Ashwini Pokle, William Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. *arXiv preprint arXiv:2406.14548*, 2024. 2

[9] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 1, 2, 3, 6, 7

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2014. 1

[11] Matej Grcić, Ivan Grubišić, and Siniša Šegvić. Densely connected normalizing flows. In *Advances in Neural Information Processing Systems*, 2021. 7

[12] Ming Gui, Johannes Schusterbauer, Ulrich Prestel, Pingchuan Ma, Dmytro Kotovenko, Olga Grebenkova, Stefan Andreas Baumann, Vincent Tao Hu, and Björn Ommer. Depthfm: Fast generative monocular depth estimation with flow matching. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(3):3203–3211, 2025. 1

[13] Pengsheng Guo and Alexander G Schwing. Variational rectified flow matching. *arXiv preprint arXiv:2502.09616*, 2025. 2

[14] Yinbin Han, Meisam Razaviyayn, and Renyuan Xu. Neural network-based score estimation in diffusion models: Optimization and generalization. In *The Twelfth International Conference on Learning Representations*, 2024. 1

[15] 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 *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 6

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020. 1

[17] Tejas Jayashankar, Jongha Jon Ryu, and Gregory W. Wornell. Score-of-mixture training: One-step generative model training made simple via score estimation of mixture distributions. In *Forty-second International Conference on Machine Learning*, 2025. 1

[18] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. 2

[19] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 7

[20] Leon Klein, Andreas Krämer, and Frank Noe. Equivariant flow matching. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1

[21] Nikita Maksimovich Kornilov, Petr Mokrov, Alexander Gasnikov, and Alexander Korotin. Optimal flow matching: Learning straight trajectories in just one step. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 1, 2

[22] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 6

[23] Dohyun Kwon, Ying Fan, and Kangwook Lee. Score-based generative modeling secretly minimizes the wasserstein distance. In *Advances in Neural Information Processing Systems*, pages 20205–20217. Curran Associates, Inc., 2022. 1

[24] Sangyun Lee, Beomsu Kim, and Jong Chul Ye. Minimizing trajectory curvature of ode-based generative models. In *International Conference on Machine Learning*, pages 18957–18973. PMLR, 2023. 2

[25] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 1, 2

[26] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1, 2, 7

[27] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and qiang liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[28] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky T. Q. Chen. Multisample flow matching: Straightening flows with minibatch couplings. In *International Conference on Machine Learning*, 2023. 1

[29] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 1

[30] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 1

[31] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*, 2024. 1, 7

[32] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 1

[33] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023. 2

[34] Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to DNA sequence design. In *ICLR 2024 Workshop on Machine Learning for Genomics Explorations*, 2024. 1

[35] 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. *Transactions on Machine Learning Research*, 2024. Expert Certification. 1

[36] Cédric Villani et al. *Optimal transport: old and new*. Springer, 2009. 2

[37] Zhisheng Xiao, Qing Yan, and Yali Amit. Generative latent flow. *arXiv preprint arXiv:1905.10485*, 2019. 7

[38] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4), 2023. 1

[39] Ling Yang, Zhaochen Yu, Chenlin Meng, Minkai Xu, Stefano Ermon, and Bin Cui. Mastering text-to-image diffusion: recaptioning, planning, and generating with multimodal llms. In *Proceedings of the 41st International Conference on Machine Learning*. JMLR.org, 2024. 1

[40] Ling Yang, Zixiang Zhang, Zhilong Zhang, Xingchao Liu, Minkai Xu, Wentao Zhang, Chenlin Meng, Stefano Ermon, and Bin Cui. Consistency flow matching: Defining straight flows with velocity consistency. *arXiv preprint arXiv:2407.02398*, 2024. 1, 2, 7

[41] Shenghao Yang and Kimon Fountoulakis. Weighted flow diffusion for local graph clustering with node attributes: an algorithm and statistical guarantees. In *Proceedings of the 40th International Conference on Machine Learning*, pages 39252–39276. PMLR, 2023. 1