COMPLEXITY-BALANCED DIFFUSION SPLITTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard continuous-time generative models rely on monolithic architectures that must navigate vastly different signal regimes, from isotropic noise to intricate data distributions. While scaling model capacity improves performance, deploying a massive network uniformly across the entire generative timeline is inherently inefficient. In this work, we propose *Complexity-Balanced Splitting* (*CBS*), a principled framework for temporal capacity allocation that distributes the generative workload across multiple specialized sub-networks. Motivated by function approximation theory and de Boor's equidistribution principle, *CBS* partitions the diffusion timeline into segments of equal approximation burden, allocating more representational capacity to regions where the generative dynamics are more difficult to model. To estimate this local complexity, we introduce two complementary and tractable monitor functions: a spatial measure based on the flow's Dirichlet energy, and a geometric measure based on the acceleration of the sampling trajectories. Using a lightweight auxiliary model to estimate these complexity profiles, our approach eliminates the need for heuristic temporal splits or computationally expensive search procedures. Extensive evaluation across multiple architectures (SiT, JiT, and UNet) and datasets demonstrates that *CBS* consistently improves synthesis quality without increasing per-step inference cost. In particular, *CBS* improves FID by ~35% on SiT-XL with CFG relative to uniform temporal partitioning.

1 INTRODUCTION

Diffusion models (Ho et al., 2020; Song et al., 2020) have established themselves as a dominant paradigm for high-fidelity generative modeling. A primary driver of this success is their remarkable scalability, which offers a reliable pathway to translate increased model capacity and compute into state-of-the-art visual synthesis (Black Forest Labs, 2024). Yet, standard diffusion frameworks rely on a *monolithic* neural architecture to execute the entire denoising process. In this setting, a single model must operate across vastly different signal regimes, ranging from near-isotropic noise to highly structured data distributions, continuously adapting its function from coarse structural formation to fine-grained refinement.

To cope with this heterogeneity, the standard practice is to scale up the model (Liang et al., 2024; Peebles & Xie, 2023). However, this strategy is inherently inefficient: the full scaled-up network is deployed uniformly across all timesteps, despite the fact that no individual denoising regime warrants such massive capacity on its own. As a more efficient alternative, one can distribute capacity temporally by training multiple specialized networks, each responsible for a different phase of the denoising process (Balaji et al., 2022; Park et al., 2023; Feng et al., 2023; Park et al., 2024; Lee et al., 2024). This design is also being used at scale, *e.g.*, by Wan 2.2 (Wan et al., 2025). Because only one sub-network is evaluated at each timestep, this design enables scaling the total parameter count without increasing per-step inference cost (FLOPs). It trades extra stored parameters and training compute for quality at fixed per-step cost, which suits models that are trained once and served many times.

A central challenge in this paradigm is determining how to partition the diffusion timeline. Existing approaches typically rely on heuristic splits (Feng et al., 2023) or computationally expensive search procedures over candidate boundaries (Balaji et al., 2022), which involve training multiple large-scale models under different partitioning schemes, most of which are ultimately discarded. Thus, these methods lack a *principled criterion* for distributing capacity over time, limiting both their efficiency and generality.

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.

In this work, we propose *Complexity-Balanced Splitting* (*CBS*), a principled framework for temporal capacity allocation in diffusion models. Our key idea is to consider the problem from a functional approximation standpoint and apply the de Boor equidistribution principle to partition the diffusion timeline into segments of equal approximation burden. Intuitively, this allocates more representational capacity to regions where the generative dynamics are more difficult to model, leading to a more uniformly accurate flow field and thus improved sample quality.

We consider two complementary monitor functions for estimating the local approximation burden of the target flow. The first is motivated by neural approximation bounds based on the spatial variation of the flow field. While existing bounds are computationally intractable in high dimensions, we relate them to the flow's Dirichlet energy, which can be estimated efficiently. The second monitor function captures the geometric complexity of the sampling trajectories induced by the flow field through their second-order time derivative, which is an established complexity measure in geometric modeling.

We evaluate our approach across multiple architectures and datasets, demonstrating consistent improvements in synthesis quality without increasing per-step inference cost. Extensive evaluation of both monitor functions shows a substantial improvement over naive time-splitting, and the derived boundaries land at the best point of their local neighborhood without a computationally prohibitive exhaustive search. Specifically, we show an improvement of ~12% without CFG (Ho & Salimans, 2022) to ~35% with CFG in FID scores on SiT-XL with respect to uniform splitting. Furthermore, both monitor functions outperform uniform splitting, with path acceleration performing best. *CBS* also outperforms a hand-designed low-noise split and boundaries placed at the analytical dynamical regimes of Biroli et al. (2024). Since unequal intervals also reweight training timesteps, we further show that *CBS*'s gains persist on top of explicit Min-SNR loss weighting (Hang et al., 2023).

2 Preliminaries

We provide here the theoretical background for our approach, specifically, we frame generative modeling as continuous-time velocity prediction and review the theory behind global function approximation and domain decomposition. We use the latter to derive our principled time-splitting scheme designed to evenly allocate the representational workload across the entire generative process.

2.1 Diffusion Models via Velocity Prediction

Diffusion models (Ho et al., 2020) and flow matching (Lipman et al., 2022) can be unified as continuous-time interpolants, in which the state x_t, $t \in [0, 1]$, bridges noise $x_0 \sim \mathcal{N}(0, I)$ and data $x_1 \sim q(x_1)$, passing from coarse structure formation at high noise to fine detail refinement near the data.

Whether formulated as flow matching (Lipman et al., 2022) or v-prediction diffusion (Salimans & Ho, 2022), modern methods train a neural network $v_\theta(x_t, t)$ to predict the trajectory's true instantaneous velocity, $u(x_t, t) = \frac{d}{dt}x_t$. The unified optimization objective is to regress the network against this ground-truth velocity,

$$\mathcal{L} = \mathbb{E}_{t,x_0,x_1}\left[\|v_\theta(x_t, t) - u(x_t, t)\|^2\right]. \tag{1}$$

By viewing the generative process through this lens, the network's only task is to approximate the local velocity field of the trajectory.

2.2 Global Function Approximation and Modeling Error

Our central challenge is that a finite-capacity neural network must approximate a target velocity field whose complexity varies substantially over time. To motivate our time-splitting strategy, we draw on classical approximation theory and domain decomposition methods, which study how approximation error should be distributed across a target domain.

Domain Decomposition and Complexity Distribution. To circumvent the limitations of global approximation, a powerful strategy is the domain splitting that we follow. Rather than fitting a single global model to u over the entire domain Ω, it is partitioned into a set of N disjoint intervals. Let $\Omega = [0, 1]$ be partitioned by a set of nodes (or knots):

$$0 = t_0 < t_1 < t_2 < \cdots < t_{N-1} < t_N = 1, \tag{2}$$

on each interval $\Omega_i = [t_{i-1}, t_i]$, we deploy a separate localized model $\hat{f}_i$. Assuming all the models are of the same form and possess the same modeling capacity, we transform the problem to optimally allocating sub-problems, *i.e.*, modeling intervals, of equal complexity. The fundamental challenge then becomes: *How should we choose the splitting points* $\{t_i\}$ *such that the total modeling complexity is optimally distributed across the domain?*

The De Boor Principle and Equidistribution. The theoretical framework for optimal domain partitioning originates in approximation theory, specifically in the study of spline approximations formulated by de Boor (1973). The core concept is that the nodes should be densely clustered in regions where the target function f is highly complex, and sparsely distributed in regions where f is smooth.

To formalize this idea, a *monitor function* $m(t) > 0$ is introduced to quantify the local approximation burden of the target function f at time $t \in [0, 1]$. Thus, regions where $m(t)$ is large require finer partitioning, while regions with small $m(t)$ can be modeled using wider intervals. In classical polynomial spline approximation of degree k, the monitor function is often chosen proportional to a fractional power of the $(k + 1)$-th derivative of f (Gallier, 1999).

Following spline approximation theory (de Boor, 1973), we adopt an *accumulation* error model, in which the modeling error on the i-th interval $[t_{i-1}, t_i]$ is proportional to the integral of the monitor function over that interval,

$$E_i \approx C \int_{t_{i-1}}^{t_i} m(t)\, dt \tag{3}$$

where C is a constant independent of the partition. For splines this is a derived bound; for neural sub-networks it is an assumption (A2 below).

The objective of minimizing the maximal approximation error across all sub-intervals, leads to a min-max optimization problem for the knot placement:

$$\min_{\{t_i\}} \max_{1 \le i \le N} \int_{t_{i-1}}^{t_i} m(t)\, dt \tag{4}$$

The De Boor Principle (de Boor, 1973) shows that this problem is solved when the error bound is *equidistributed* across all intervals. In other words, the minimizer of this surrogate is achieved when the integral of the monitor function is strictly equal for every sub-interval, yielding the final equidistribution,

$$\int_{t_{i-1}}^{t_i} m(t)\, dt = \frac{1}{N} \int_0^1 m(t)\, dt \quad \text{for all } i = 1, 2, \ldots, N \tag{5}$$

As a result, each local model faces a comparable representational burden which aligns well with having the same modeling capacity. This principle forms the foundation of our approach to temporal capacity allocation in diffusion models.

Assumptions and scope. We assume **(A1)** equal-capacity sub-networks, **(A2)** the accumulation model above, and **(A3)** that our monitors (Sec. 3) are faithful proxies for a deep network's learning burden. Given (A1)–(A2), equidistribution exactly minimizes the min–max surrogate for *any finite* N, with no limiting argument; (A3) cannot be proven for modern architectures and is tested empirically (Sec. 4). The theory thus motivates the principle and the form of the monitor, not an FID-optimal split.

Finally, let us consider our specific context where the target function u is a flow field which is path-integrated during sampling generation. In this case, once a large error is encountered, the resulting path diverges from its true course and the misalignment is maintained forward in time, even if the flow is fairly accurate in the subsequent steps, as illustrated in Fig. 1. Grönwall's inequality (Gronwall, 1919) bounds the terminal error by the *maximal* local velocity error, motivating control of the worst interval; our surrogate is a proxy for, not a bound on, that error.



Figure 1: Maximal local errors govern ODE approximation.

In this respect we note that standard training paradigms for denoising models, such as DDPM (Ho et al., 2020) and flow matching (Lipman et al., 2022), optimize for the expected error averaged across

the entire denoising interval (as seen in Eq. 1). In our framework the two act at different levels: each sub-network is trained with the expected loss of Eq. 1 on its interval, while the min–max criterion only governs the *outer* choice of the partition.

3 METHOD

To derive our time-splitting scheme based on these principles, we must define a meaningful monitor function $m(t)$. We propose two such functions: the first assesses the target flow field's complexity, via Barron's error bound (Barron, 1993) (Sec. 3.1), while the second evaluates the geometric complexity of the resulting flow paths (sampling trajectories) via classical curve approximation measures (Sec. 3.2). These two approaches offer complementary views of the analytical complexity of a generative flow field: a spatial one and a temporal (trajectory-based) one. Finally, in Sec. 3.4 we describe our training and inference pipelines.

3.1 MODELING BURDEN VIA DIRICHLET SPECTRAL ENERGY

Barron's theorem (Barron, 1993) relates the approximation error of a network to the spectral properties of the target function. Formally, for a target function f defined on a bounded domain of radius r, and data drawn from a probability distribution p_t, a feedforward network f_n with n hidden units achieves an expected Mean Squared Error (MSE) bounded by:

$$\varepsilon_{f_n} = \mathbb{E}_{p_t}\left[\|f(x) - f_n(x)\|^2\right] \leq \frac{4r^2 C_f^2}{n}, \tag{6}$$

where C_f is the spectral complexity of the target function, formally defined as the first absolute moment of the function's Fourier transform:

$$C_f = \int_{\mathbb{R}^d} \|\omega\| |\hat{f}(\omega)| d\omega. \tag{7}$$

A smooth, gently changing function possesses a low C_f, whereas a highly non-linear function exhibiting sharp transitions is characterized by a higher C_f. This is an existence bound for one-hidden-layer sigmoidal networks on a bounded domain; it does not bound the optimization error of the transformers and UNets we train, so we use it to motivate *which quantity* to monitor, not as an architecture-specific error bound.

In our context of modeling denoising flows, the target function is the instantaneous vector field, $v_t(x)$, discussed in Sec. 2.1. Consequently, we cannot assess C_{v_t} in practice due to the flow's exceedingly high spatial dimension. Instead, we obtain a formal bound on C_{v_t} based on the vector field's global spatial variation, measured by its *Dirichlet energy*

$$E_D(v_t) = \frac{1}{2}\int_{\mathbb{R}^d} \|\nabla_x v_t(x)\|^2 dx. \tag{8}$$

Intuitively, rapid spatial variation is associated with high spectral energy: via Parseval's identity (Rudin, 1964) and the Cauchy–Schwarz inequality over an effective frequency bandwidth (App. A), $C_{v_t}^2 \leq K_t\, E_D(v_t)$, where K_t depends on the support of the field's spectrum. Since that spectrum varies with t, so does K_t; we treat it as approximately constant (a worst case), so that temporal variation in the bound is carried by $E_D(v_t)$. Substituting this bound back into Barron's theorem yields a direct, computable monitor function based on the global spatial roughness of the flow and the expected approximation error for a capacity budget n:

$$\varepsilon_{f_n} \leq K' \frac{E_D(v_t)}{n} = m(t). \tag{9}$$

The full derivation of this bound, including the formulation of the bandwidth constant K_t, is provided in Appendix A. In practice, our estimator (Sec. 3.4) targets the density-weighted energy $E_D^{p_t}(v_t) = \frac{1}{2}\mathbb{E}_{x\sim p_t}\|\nabla_x v_t(x)\|_F^2$ rather than Eq. 8; since the Parseval step applies only to the unweighted form, this is a motivated approximation rather than a bound (App. A).

3.2 Modeling Burden via Path Acceleration

While the Dirichlet energy bounds the modeling burden from the perspective of the spatial flow field complexity, we explore an alternative and complementary bound that we derive by analyzing the *temporal complexity* of the underlying sampling trajectories.

For any given initial noise state x_0, the flow field u defines a continuous trajectory x_t in $\mathbb{R}^d$, as a function of time $t \in [0, 1]$. The modeled flow v_θ gives rise to an approximate trajectory $\hat{x}_t$, thus modeling errors between the two naturally arise. The approximation theory literature, ranging from classic n-degree polynomials or Fourier series to recent specific classes of deep neural networks (Yarotsky, 2017), encounters a general bound on the global approximation error on curves, given by

$$\|x_t - \hat{x}_t\|_\infty \leq C \frac{L_k}{n^k} \tag{10}$$

where n represents the model capacity (parameter count), k denotes the order of the derivative used to assess the trajectory's smoothness, and the constant C is independent of n and the trajectory. In this global bound, the curve's k-th derivative term L_k is maximized over the entire interval, by $L_k = \sup_t \left\| d^k x_t / dt^k \right\|$, and is therefore overly pessimistic as it may vary substantially over time.

Following the accumulation model (A2) of Sec. 2.2, we instead employ the local magnitude of the k-th derivative as a timewise monitor function and integrate it over each time segment, rather than taking its supremum; the splitting then follows from the equidistribution condition, Eq. 5.

Larger k yields faster asymptotic convergence but is harder to estimate reliably, while $k{=}1$ bounds the error by the path's velocity, which conflates geometric complexity with traversal speed and penalizes displacement rather than non-linearity. Instead, as a monitor function we opt for $k = 2$, which bounds the error using the path's acceleration,

$$m(t) = \left\| \frac{d^2 x_t}{dt^2} \right\|, \tag{11}$$

which effectively filters out the constant-velocity displacement and isolates the curviness of the path. In case of trajectories with relatively constant velocity magnitudes, the second-order derivative effectively approximates the curvature, a well-established measure of complexity in classical geometric modeling (Gallier, 1999).

Relation to temporal variation. Since $dx_t/dt = v_t(x_t)$, the chain rule gives

$$\tfrac{d^2 x_t}{dt^2} = \partial_t v_t(x_t) + \nabla_x v_t(x_t)\, v_t(x_t). \tag{12}$$

The acceleration thus captures both the explicit temporal variation of the field, which one sub-network must absorb across its whole interval, and a Jacobian (spatial) term; the pointwise-in-t Dirichlet monitor captures only the latter. For perfectly straight trajectories the monitor is constant and ***CBS*** reduces to uniform splitting.

3.3 Deriving the Time-Splitting Scheme

Both monitor functions $m(t)$ discussed above require access to the target flow field and its induced trajectories. To obtain these, we first train a single auxiliary network over the entire time frame $t \in [0, 1]$. Because this network is used solely for estimating the temporal boundaries t_i from Eq. 5, a highly approximate model is sufficient. By training a smaller architecture on a fraction of the dataset (10% in our implementation) for fewer epochs, we obtain a sufficiently reliable time-splitting with minimal overhead, as validated in App. C.

Using this auxiliary flow network, we generate K sampling trajectories, $\{x_t^k\}_{k=1}^K$, which serve for estimating the monitor functions $m(t)$ over a uniform temporal grid (100 grid points in our implementation).

Dirichlet Energy Monitor Function. Evaluating $E_D(v_t)$ is computationally demanding because the spatial gradient term $\|\nabla_x v_t(x)\|^2$ represents the squared Frobenius norm of the $d \times d$ Jacobian matrix, and d is exceedingly large in practice. To bypass materializing the full Jacobian, we employ randomized trace estimators (Hutchinson, 1990) that rely on efficient Jacobian-Vector Products

(JVPs). At each grid point t, our K trajectories provide samples from the marginal distribution p_t. Averaging the JVP-based gradient estimates across these K points estimates the density-weighted energy (Eq. 21), which is the monitor we use.

Path Acceleration Monitor Function. The K trajectories are evaluated on a discrete temporal grid. We approximate the second-order time derivative via a first-order finite difference over the velocity field v_t. Specifically:

$$m(t) = \frac{1}{K\,\Delta t} \sum_{k=1}^{K} \|v_{t+\Delta t}(x^k_{t+\Delta t}) - v_t(x^k_t)\|, \tag{13}$$

which follows from the relation $dx_t/dt = v_t(x_t)$.

Finally, given $m(t)$ evaluated across the uniform temporal grid, we approximate de Boor's equidistribution principle discretely to determine the final time partition points, t_i. Specifically, we compute the cumulative sum of $m(t)$ and select the boundaries as the grid points that partition the total accumulated monitor value equally among the N segments.

Formally, the i-th split t_i is chosen as the grid point where the cumulative sum most closely approximates $\frac{i}{N}\sum_j m(t_j)$.

We compute the monitor once from the unguided field and share the split across all CFG scales; recomputing it on the guided field shifts the boundaries by less than 0.01.

3.4 Training and Inference

The resulting t_i are used as the boundaries in both training and inference of *CBS* as elaborated next.

Training. During the optimization phase, each specialized network v_{θ_i} is trained independently using the standard velocity prediction objective (Eq. 1). However, its time domain is strictly bounded: the timestep t is sampled exclusively from its designated interval, as summarized in Alg. 1.

Inference. Generating a novel sample is done by switching between the networks, using each one v_{θ_i} over its designated segment $[t_{i-1}, t_i]$. Per-step FLOPs, and peak memory when experts are loaded sequentially, match a single network.

4 Experiments

We implemented and evaluated *CBS* to assess its performance, robustness and generality. Since our monitor functions are theory-motivated proxies rather than provable bounds on a deep network's error (Sec. 2.2), our evaluation aims to test empirically whether equidistributing them yields good splits across multiple architectures, datasets, and numbers of partitions. We start by detailing our experimental settings across three generative domains (Sec. 4.1). We then present our time-splitting performance in terms of added generative accuracy (Sec. 4.2). This is followed by a sensitivity analysis of the boundaries, a comparison with hand-designed splits, and a direct comparison between our proposed monitor functions (Sec. 4.3). We then separate the effect of specialization from that of timestep reweighting (Sec. 4.4). Finally, we show that these temporal boundaries can be estimated with negligible computational cost (App. C).

4.1 Experimental Setup

We evaluate *CBS* in three settings: class-conditional latent synthesis on ImageNet-256 with SiT (Ma et al., 2024) (S, B, and XL), pixel-space synthesis on ImageNet-64 with JiT (Li & He, 2025), and unconditional CIFAR-10 with a convolutional UNet (Ronneberger et al., 2015). We report FID (Heusel et al., 2017), Inception Score (Salimans et al., 2016), and Precision/Recall (Kynkäänniemi et al., 2019); full descriptions are given in App. G.

Implementation Details. Unless otherwise specified, all experiments use $N = 3$ specialized networks, and the path acceleration based monitor function $m(t)$. All monolithic baselines and our sub-networks are trained using the same default hyperparameters to ensure a fair comparison; each sub-network is trained for the same number of iterations as the monolithic baseline. Following standard academic practice (Peebles & Xie, 2023; Ma et al., 2024), we use a 400K-iteration budget on ImageNet-256, so all results are controlled comparisons against identically trained baselines

Table 1: **Quantitative evaluation of SiT on ImageNet-256.** We compare the standard monolithic baseline, a naive uniformly partitioned ensemble (partitioned at $0.33, 0.66$), and our approach (*CBS*) using complexity-derived boundaries $(0.4, 0.77)$. Across all model capacities (S/2, B/2, and XL/2), our method yields consistent and significant improvements in FID, Inception Score (IS), and Precision, with Recall on par or better, both with and without Classifier-Free Guidance (CFG). Crucially, *CBS* achieves these generative gains while strictly maintaining the exact same active parameter count and per-step inference cost (GFLOPs) as the standard monolithic baseline, while storing $N\times$ the total parameters. For SiT-B/2 we also report a hand-designed non-uniform split that favors the low-noise regime (0.5, 0.8).

			Without CFG				With CFG			
Model	**# Act. Par.**	**GFLOPs**	**FID↓**	**IS↑**	**Prec.↑**	**Rec.↑**	**FID↓**	**IS↑**	**Prec.↑**	**Rec.↑**
SiT-S/2	33	6.06	58.97	23.34	0.40	0.59	30.10	49.82	0.57	0.50
SiT-S/2 (0.33 0.66)	33	6.06	58.82	23.23	0.40	0.59	29.83	52.17	0.57	**0.51**
SiT-S/2 Ours (0.4 0.77)	33	6.06	**50.87**	**29.57**	**0.44**	**0.62**	**18.61**	**72.73**	**0.62**	**0.51**
SiT-B/2	130	23.01	34.84	41.53	0.52	0.64	16.79	84.75	0.66	0.55
SiT-B/2 (0.33 0.66)	130	23.01	34.52	41.45	0.52	0.64	16.51	87.21	0.66	**0.56**
SiT-B/2 (0.5 0.8)	130	23.01	32.04	47.51	0.54	0.64	12.93	105.03	0.69	**0.56**
SiT-B/2 Ours (0.4 0.77)	130	23.01	**30.51**	**52.20**	**0.55**	**0.65**	**10.72**	**121.09**	**0.72**	**0.56**
SiT-XL/2	675	118.64	18.04	73.90	0.63	0.64	6.53	162.23	0.73	**0.56**
SiT-XL/2 (0.33 0.66)	675	118.64	17.97	73.85	0.63	0.64	6.24	165.29	0.74	**0.56**
SiT-XL/2 Ours (0.4 0.77)	675	118.64	**15.81**	**86.43**	**0.64**	**0.66**	**4.03**	**195.72**	**0.80**	**0.56**

Table 2: **Quantitative evaluation of JiT-B/4 on ImageNet-64.** In order to verify that *CBS* generalizes effectively to raw, pixel-space spatial gradients, we compare our complexity-based partitioning against the standard monolithic JiT-B/4 architecture and a uniform temporal split. All configurations maintain an identical inference cost of 131M activated parameters and 25 per-step GFLOPs. Even with this strict compute budget, our method achieves significant improvements in synthesis quality.

	Without CFG				With CFG			
Model	**FID↓**	**IS↑**	**Prec.↑**	**Rec.↑**	**FID↓**	**IS↑**	**Prec.↑**	**Rec.↑**
JiT-B/4	17.43	25.92	0.55	0.59	16.41	158.52	0.87	0.28
JiT-B/4 (0.33 0.66)	16.39	28.04	0.57	**0.61**	15.08	179.39	0.88	0.28
JiT-B/4 Ours (0.1 0.38)	**15.02**	**29.86**	**0.59**	**0.61**	**13.93**	**199.38**	**0.89**	**0.30**

rather than state-of-the-art numbers, which require several million iterations. The cumulative monitor energy used to derive our time-splits is pre-computed over a grid of 100 points, as described in App. C. The exact training hyperparameters, sampling configurations, and further hardware specifics are detailed in Appendix H.

4.2 Generative Performance and Network Scaling

Latent and Pixel-Space Synthesis. To evaluate the core efficacy of *CBS*, we compare our multi-network configuration against standard monolithic models. As demonstrated in Tab. 1, splitting the generative workload according to our complexity monitor functions on the latent ImageNet-256 allows our partitioned SiT architecture to achieve significantly superior synthesis quality (measured via FID and IS) without inflating the per-step inference FLOPs. Furthermore, we observe that these architectural benefits compound significantly when utilizing Classifier-Free Guidance (CFG). We hypothesize that the localized capacity helps resolve the complex spatial gradients introduced by the guidance term, although our splits are derived from the unguided field (Sec. 3.3).

Table 3: **Unconditional generation on CIFAR-10.** Regime: boundaries at the dynamical regimes of Biroli et al. (2024).

Model	**FID↓**
Baseline	3.55
(0.33 0.66)	3.52
Regime (0.15, 0.82)	2.96
Ours (0.2, 0.78)	**2.72**

This performance generalizes beyond flows in latent space. As shown in Tab. 2, *CBS* also achieves significant improvements over the monolithic baseline in pixel-space ImageNet-64 with the JiT architecture. Finally, our UNet-based evaluation on unconditional CIFAR-10 (Tab. 3) confirms that the method scales down effectively to smaller datasets and non-transformer architectures. Using more networks ($N{=}1$ to 4) consistently improves FID and IS (App. D).

Table 4: **Sensitivity to boundary placement** on SiT-B/2, perturbing our derived splits $(0.4, 0.77)$.

Model	FID↓	IS↑	Prec.↑	Rec.↑
SiT-B/2 (0.35 0.77)	31.44	49.83	0.54	**0.65**
SiT-B/2 (0.45 0.77)	31.21	50.11	0.54	0.64
SiT-B/2 (0.4 0.72)	30.89	51.17	0.54	0.64
SiT-B/2 (0.4 0.82)	31.54	50.13	0.54	0.64
SiT-B/2 Ours (0.4 0.77)	**30.51**	**52.20**	**0.55**	**0.65**

Table 5: **Specialization vs. reweighting** (SiT-B/2, FID↓ w/o / w/ CFG). Right: all models trained with Min-SNR loss weighting (Hang et al., 2023).

	Standard loss		Min-SNR loss	
Monolithic	34.84	16.79	26.13	13.32
Uniform split	34.52	16.51	25.92	12.97
CBS	**30.51**	**10.72**	**23.36**	**8.72**

4.3 Sensitivity to Boundary Placement

Since our split is derived from a surrogate (Sec. 2.2) rather than from FID, we compare it with nearby alternatives. This is done by training multiple sets of networks over perturbed versions of our temporal splits (changing each boundary separately).

As shown in Tab. 4, matching the temporal splits exactly to intervals of equal cumulative complexity (our time-splitting) consistently yields the lowest FID. The gaps are moderate (0.4–1.0 FID), so the derived split is the best in its local neighborhood.

Hand-designed and regime-based splits. A hand-designed low-noise split, $(0.5, 0.8)$, reaches 32.04 / 12.93 FID on SiT-B/2 (w/o / w/ CFG) vs. 30.51 / 10.72 for *CBS* (Tab. 1). Boundaries at the dynamical regimes of Biroli et al. (2024), which characterize the data rather than the learned flow, yield 2.96 vs. 2.72 FID on CIFAR-10 (Tab. 3).

Comparison of monitor functions. Both monitor functions yield splits that outperform uniform splitting, and the path-acceleration monitor obtains better FID on both SiT-B/2 and JiT-B/4, so we use it as our default (App. E).

4.4 Specialization vs. Timestep Reweighting

Since each sub-network is trained on its own interval for the full schedule, the ensemble jointly sees a piecewise-uniform timestep density $1/\big(N(t_i - t_{i-1})\big)$ on interval i ($\approx 0.83, 0.90, 1.45$ for our ImageNet split), so its gains could stem from reweighting rather than specialization. In Tab. 5, the uniform split, whose induced timestep density is exactly uniform, yields only a small gain, showing that the benefit depends on where the boundaries are placed. When all configurations are combined with the explicit Min-SNR weighting (Hang et al., 2023), *CBS* remains best, with a relative gain comparable to that under the standard loss (35% vs. 36% with CFG), indicating that the gains are not explained by generic loss reweighting.

5 Related Work

Temporal Specialization in Diffusion Models. Recognizing that global networks struggle to efficiently model heterogeneous generative trajectories, several works explore temporal specialization. Cascaded models (Ho et al., 2022) divide generation across independent networks, but partition by spatial resolution rather than time. Operating explicitly on the time axis, eDiff-I (Balaji et al., 2022) and MEME (Lee et al., 2024) train expert denoisers for specific noise intervals, yet identifying good transition boundaries requires exhaustively expensive empirical searches. At scale, Wan 2.2 (Wan et al., 2025) deploys two noise-level experts switched at a signal-to-noise threshold, a setting in which a principled boundary criterion is directly applicable. Other approaches use time-conditioned Mixture-of-Experts (MoE) and dynamic task routing to bypass strict boundaries. Models like Denoising Task Routing (DTR) (Park et al., 2023), Switch Diffusion Transformers (Park et al., 2024), RAPHAEL (Xue et al., 2023), and recent MoE-transformers (Cheng et al., 2025) dynamically allocate compute based on the active timestep. While effective, these learned, black-box routing mechanisms are notoriously difficult to stabilize, prone to routing collapse, and lack guarantees for balanced representational work. In contrast, our method provides a mathematically principled, search-free criterion to partition the timeline.

Approximation Theory in Neural Networks. Approximation theory provides rigorous bounds on the representational capacity required to fit complex functions. Classic theorems by Barron

(1993) bound feedforward network errors using the target function's spectral complexity, formally linking high-frequency spatial fluctuations to larger required parameter counts. For deep architectures, Yarotsky (2017) extended these bounds to deep ReLU networks approximating functions in Sobolev spaces, defining error decay rates governed by the magnitude of high-order derivatives. Beyond static functions, continuous vector field representation has been analyzed in Neural ODEs, bounding trajectory complexity and integration error via Jacobian traces and Hutchinson estimators (Finlay et al., 2020; Kelly et al., 2020; Hutchinson, 1990). While profound, applying these frameworks to generative models is typically hindered by the intractability of evaluating high-dimensional spectral norms or exact high-order derivatives. Our work bridges this gap by using these bounds to motivate tractable monitor functions. By relating spectral complexity to Dirichlet energy and trajectory error to path acceleration, we turn classical approximation theory into a practical tool for allocating network capacity over time.

Timestep Weighting and Denoising Regimes. Another line of work keeps a single network and reweights or resamples timesteps during training (Choi et al., 2022; Karras et al., 2022; Esser et al., 2024; Hang et al., 2023). These methods redistribute the *optimization effort* of a fixed-capacity network, whereas *CBS* redistributes *capacity*, and the two are complementary (Sec. 4.4). Biroli et al. (2024) derive dynamical regimes of the backward diffusion analytically from the data distribution; we compare against boundaries placed at these regimes in Sec. 4.3.

Scaling Up Generative Models. The predictable improvement of neural networks with increased capacity, formalized as scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022; Liang et al., 2024), remains foundational in deep learning. In visual generative modeling, this principle has driven a push toward larger architectures. Breakthroughs like the Diffusion Transformer (DiT) (Peebles & Xie, 2023), Scalable Interpolant Transformers (SiT) (Ma et al., 2024), and modern large-scale models (Black Forest Labs, 2024; Wu et al., 2025) demonstrate that aggressively scaling parameters yields predictable improvements in sample fidelity. However, standard continuous-time frameworks deploy monolithic architectures, applying this massive parameter budget at every integration step. Consequently, adhering to scaling laws incurs a proportional and often prohibitive increase in inference-time computational cost (FLOPs). Our approach decouples parameter scaling from inference cost: by distributing the expanded capacity across the temporal axis, we allow total model capacity to grow while ensuring the active parameter count at any given timestep remains constant.

6 Conclusion

We introduced Complexity-Balanced Splitting (*CBS*), a principled framework for temporal capacity allocation in continuous-time generative models. To overcome the inefficiency of scaling monolithic architectures, we framed timeline partitioning as a domain decomposition problem in the context of approximation theory. Leveraging de Boor's equidistribution principle, we showed empirically that dividing the timeline into segments of equal estimated burden, measured using either spatial Dirichlet energy or temporal path acceleration, outperforms uniform, hand-designed, and regime-based splits. While path acceleration proved to be slightly advantageous, by presenting and evaluating both, we cover the two central approaches to monitor the complexity of a flow field in terms of its analytical regularity.

Empirical evaluations across diverse architectures (SiT, JiT, UNet) confirm that *CBS* significantly improves synthesis quality without increasing per-step inference costs, and that its gains persist on top of explicit timestep reweighting. Furthermore, because these complexity metrics reflect geometric properties of the generative trajectory, the boundaries can be estimated pre-training with negligible overhead, removing the need for costly empirical searches.

This work focuses on the temporal axis, where our derived splits are the best in their local neighborhood. However, we believe this playground should be expanded, and the equidistribution principle should be used to derive other forms of splitting in new domains. For instance, spatial splitting, in which individual tokens are routed between networks, represents a potentially stronger direction. Deriving the proper monitor functions for this is expected to be a challenge, and we leave this topic for future work.

Ultimately, *CBS* provides a principled, search-free solution to decouple total parameter capacity from inference costs, enabling massively scaled models that focus compute where the generative dynamics demand it most.

AI USE STATEMENT

In this work, we used generative AI tools for assistance with code implementation. We have not used generative AI tools for the development of the theoretical model, the mathematical formulations and derivations, hypothesis refinement, the design of the methodology or experiments, qualitative analysis, or the interpretation of results. Synthetic dataset generation, translation, and data cleaning are not applicable to this work. Additionally, we used generative AI tools for editing the manuscript for readability and grammar, and for formatting references.

ETHICS STATEMENT

This work proposes a method for allocating model capacity across the timeline of diffusion and flow-based generative models; it does not involve human subjects, crowdsourcing, or the collection of new data. All experiments use established, publicly available research benchmarks (ImageNet and CIFAR-10), which we use in accordance with their terms of use for non-commercial research, and we do not release any new dataset.

REPRODUCIBILITY STATEMENT

The theoretical motivation, including the equidistribution condition, is given in Sec. 2.2, with the full derivation of the Dirichlet-energy bound and its assumptions in App. A. The monitor-function estimators, boundary selection, and training and inference procedures are described in Sec. 3 and Alg. 1; the auxiliary model is specified in App. C, and the resulting boundaries are reported in App. B. Datasets, architectures, and metrics are detailed in Sec. 4.1 and App. G, and all hyperparameters and hardware details in App. H (Tab. 8). Code and trained models will be released upon acceptance.

REFERENCES

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, et al. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. Inf. Theory*, 39:930–945, 1993. URL `https://api.semanticscholar.org/CorpusID:15383918`.

Giulio Biroli, Tony Bonnaire, Valentin De Bortoli, and Marc Mézard. Dynamical regimes of diffusion models. *Nature Communications*, 15(1):9957, 2024.

Black Forest Labs. FLUX. `https://github.com/black-forest-labs/flux`, 2024.

Kun Cheng, Xiao He, Lei Yu, Zhijun Tu, Mingrui Zhu, Nannan Wang, Xinbo Gao, and Jie Hu. Diff-moe: Diffusion transformer with time-aware and space-adaptive experts. In *Forty-second International Conference on Machine Learning*, 2025.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11462–11471. IEEE, 2022.

Carl de Boor. Good approximation by splines with variable knots. In Amram Meir and Ambikeshwar Sharma (eds.), *Spline Functions and Approximation Theory*, volume 21 of *International Series of Numerical Mathematics / ISNM*, pp. 57–72. Birkhäuser Basel, 1973. doi: 10.1007/978-3-0348-5979-0_3.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10135–10145, 2023.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman. How to train your neural ode: the world of jacobian and kinetic regularization. In *International conference on machine learning*, pp. 3154–3164. PMLR, 2020.

Jean Gallier. *Curves and surfaces in geometric modeling: theory and algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999. ISBN 1558605991.

Thomas H Gronwall. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Mathematics*, 20(4):292–296, 1919.

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 *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7407–7417. IEEE, 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. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022. URL `http://jmlr.org`.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, DDL Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 10, 2022.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Jacob Kelly, Jesse Bettencourt, Matthew J Johnson, and David K Duvenaud. Learning differential equations that are easy to solve. *Advances in Neural Information Processing Systems*, 33:4370–4380, 2020.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019.

Yunsung Lee, JinYoung Kim, Hyojun Go, Myeongho Jeong, Shinhyeok Oh, and Seungtaek Choi. Multi-architecture multi-expert diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13427–13436, 2024.

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

Zhengyang Liang, Hao He, Ceyuan Yang, and Bo Dai. Scaling laws for diffusion transformers. *arXiv preprint arXiv:2410.08184*, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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 *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Byeongjun Park, Sangmin Woo, Hyojun Go, Jin-Young Kim, and Changick Kim. Denoising task routing for diffusion models. *arXiv preprint arXiv:2310.07138*, 2023.

Byeongjun Park, Hyojun Go, Jin-Young Kim, Sangmin Woo, Seokil Ham, and Changick Kim. Switch diffusion transformer: Synergizing denoising tasks with sparse mixture-of-experts. In *European Conference on Computer Vision*, pp. 461–477. Springer, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

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*, pp. 234–241. Springer, 2015.

Walter Rudin. *Principles of mathematical analysis*. 1964. URL https://api.semanticscholar.org/CorpusID:50742905.

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

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Zeyue Xue, Guanglu Song, Qiushan Guo, Boxiao Liu, Zhuofan Zong, Yu Liu, and Ping Luo. Raphael: Text-to-image generation via large mixture of diffusion paths. *Advances in Neural Information Processing Systems*, 36:41693–41706, 2023.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 94: 103–114, 2017. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.07.002. URL https://www.sciencedirect.com/science/article/pii/S0893608017301545.

A Bounding Spectral Complexity via Dirichlet Energy

In this section, we formally derive the relationship between the spectral complexity C_{v_t} and the Dirichlet energy $E_D(v_t)$ using the Cauchy-Schwarz inequality.

Recall the definition of the spectral complexity for the vector field v_t is given by

$$C_{v_t} = \int_{\mathbb{R}^d} \|\omega\| \|\hat{v}_t(\omega)\| d\omega. \tag{14}$$

To bound this integral using the L^2 norm of the gradient (which corresponds to the Dirichlet energy), we must account for the fact that the L^1 norm of a function over $\mathbb{R}^d$ cannot be bounded by its L^2 norm without an additional decaying weight or an assumption of bounded support. In practical physical and neural network applications, we assume the flow field is effectively band-limited. That is, its spectral energy is negligible outside a frequency ball of radius Ω_{max}, allowing us to restrict our domain of integration to $B(0, \Omega_{\text{max}})$. This assumption follows from smoothness: if v_t is k times continuously differentiable with integrable derivatives, its Fourier transform decays as $O(\|\omega\|^{-k})$, so truncating the spectrum at a finite radius incurs a small error. We note that the resulting constant $V(\Omega_{\max})$ grows with the dimension d, and $\Omega_{\max}$ may depend on t; the bound is therefore used only to motivate the monitor (up to the time-dependent constant K_t discussed in Sec. 3.1), not as a quantitative estimate.

The Cauchy-Schwarz inequality for integrals states:

$$\left(\int_D f(\omega) g(\omega) d\omega \right)^2 \leq \left(\int_D f(\omega)^2 d\omega \right) \left(\int_D g(\omega)^2 d\omega \right). \tag{15}$$

We set $f(\omega) = 1$ and $g(\omega) = \|\omega\| \|\hat{v}_t(\omega)\|$ over the domain $D = B(0, \Omega_{\text{max}})$. Applying the inequality yields

$$C_{v_t}^2 = \left(\int_{B(0,\Omega_{\text{max}})} 1 \cdot \|\omega\| \|\hat{v}_t(\omega)\| d\omega \right)^2 \leq \left(\int_{B(0,\Omega_{\text{max}})} 1 d\omega \right) \left(\int_{B(0,\Omega_{\text{max}})} \|\omega\|^2 \|\hat{v}_t(\omega)\|^2 d\omega \right). \tag{16}$$

The first term on the right-hand side is simply the volume of the d-dimensional ball of radius Ω_{max}, which we denote as $V(\Omega_{\text{max}})$. Assuming the energy outside this ball is negligible, the second term can be extended back to $\mathbb{R}^d$ and related to the Dirichlet energy via Parseval's identity.

Recall Parseval's identity for the gradient of a function is given by

$$\int_{\mathbb{R}^d} \|\nabla_x v_t(x)\|^2 dx = (2\pi)^d \int_{\mathbb{R}^d} \|\omega\|^2 \|\hat{v}_t(\omega)\|^2 d\omega. \tag{17}$$

Substituting the definition of the Dirichlet energy, $E_D(v_t) = \frac{1}{2} \int_{\mathbb{R}^d} \|\nabla_x v_t(x)\|^2 dx$, we have:

$$\int_{\mathbb{R}^d} \|\omega\|^2 \|\hat{v}_t(\omega)\|^2 d\omega = \frac{2}{(2\pi)^d} E_D(v_t). \tag{18}$$

Plugging this equivalence back into our Cauchy-Schwarz bound, we arrive at the final inequality,

$$C_{v_t}^2 \leq V(\Omega_{\text{max}}) \frac{2}{(2\pi)^d} E_D(v_t). \tag{19}$$

Defining the constant $K = \frac{2V(\Omega_{\text{max}})}{(2\pi)^d}$, we obtain the relation used in the main text,

$$C_{v_t}^2 \leq K \cdot E_D(v_t). \tag{20}$$

Unweighted vs. density-weighted energy. The derivation above concerns the unweighted energy $E_D(v_t)$ of Eq. 8. Our estimator, which averages Jacobian norms over samples $x \sim p_t$, targets the density-weighted energy

$$E_D^{p_t}(v_t) = \tfrac{1}{2} \, \mathbb{E}_{x \sim p_t} \left[\|\nabla_x v_t(x)\|_F^2 \right]. \tag{21}$$

The two coincide (up to a constant) only when p_t is uniform on a bounded domain, and the Parseval step does not transfer to the weighted quantity without an additional weighted analysis. We therefore present the weighted monitor as a motivated approximation. Since Barron's error (Eq. 6) is itself an expectation under p_t, variation of v_t in regions of negligible probability does not affect it, which makes the weighted energy the more relevant measure in practice. It also coincides with the Jacobian-norm regularizers used for Neural ODEs (Finlay et al., 2020; Kelly et al., 2020).

B Cumulative Path Acceleration Analysis

To further illustrate the temporal dynamics of the generative process and support the empirical findings in Section 4, Figure 2 visualizes the cumulative path acceleration (derived in Section 3.2) across our three evaluated baselines: SiT on latent ImageNet-256, JiT on pixel-space ImageNet-64, and UNet on CIFAR-10.

As observed in the plots, the accumulation of geometric trajectory acceleration is distinctly non-linear across all architectures and data modalities. By defining our temporal boundaries at intervals of equal accumulated acceleration, ***CBS*** naturally adapts to these dataset-specific dynamics. It assigns narrower time intervals (and thus, higher localized parameter capacity) to the steepest phases of the curve. This illustrates why the derived splits differ markedly from uniform ones, and why the split is dataset-dependent ($0.4/0.77$ for SiT, $0.1/0.38$ for JiT, $0.2/0.78$ for UNet).

C Efficiency of Time-Splitting Estimation

A potential practical concern regarding ***CBS*** is the reliance on a pre-trained auxiliary model to compute the cumulative modeling complexity across the generative timeline. However, we demonstrate that deriving the time partitioning with our approach incurs negligible computational overhead in practice. To validate this, we estimate the complexity with four increasingly lightweight auxiliary models: a fully trained SiT-XL, a fully trained SiT-S/2, a SiT-S/2 trained for 50K instead of 400K iterations, and a SiT-S/2 trained on 10% of ImageNet.

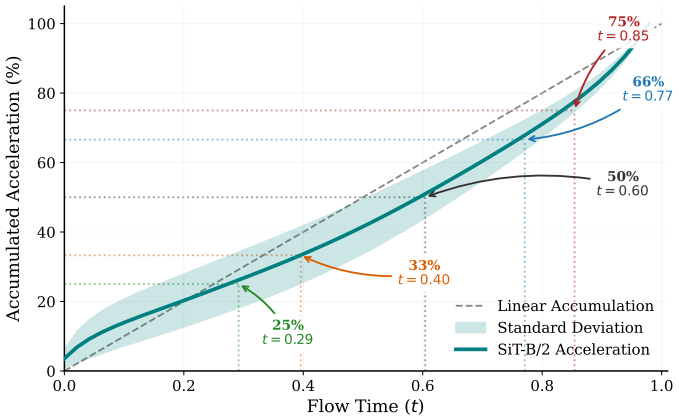
Remarkably, all four configurations yield nearly identical complexity curves and produce the exact same temporal boundary placements. This demonstrates that the flow dynamics captured by our complexity measures are largely robust to architectural scale and training duration, particularly when the objective is to assess a small number of networks (N).

D Scaling the Number of Networks

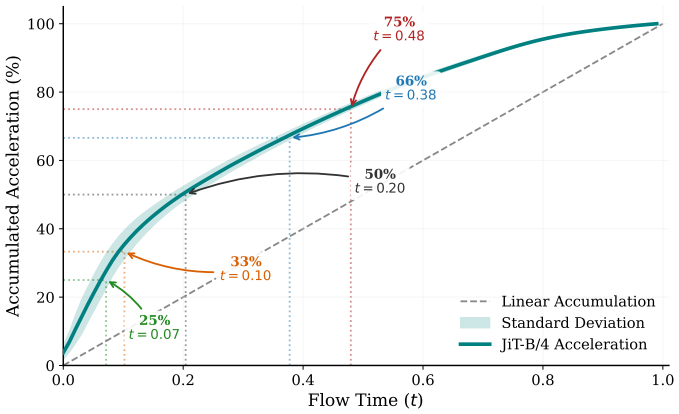
Table 6: **Scaling *CBS* across multiple networks.** We evaluate SiT-B/2 partitioned into N networks using our complexity-derived boundaries.

	Without CFG		**With CFG**	
N **Models**	**FID**↓	**IS**↑	**FID**↓	**IS**↑
$N = 1$	34.84	41.53	16.79	84.75
$N = 2$	32.19	47.82	12.98	109.42
$N = 3$	30.51	52.20	10.72	121.09
$N = 4$	29.33	55.03	9.42	129.49

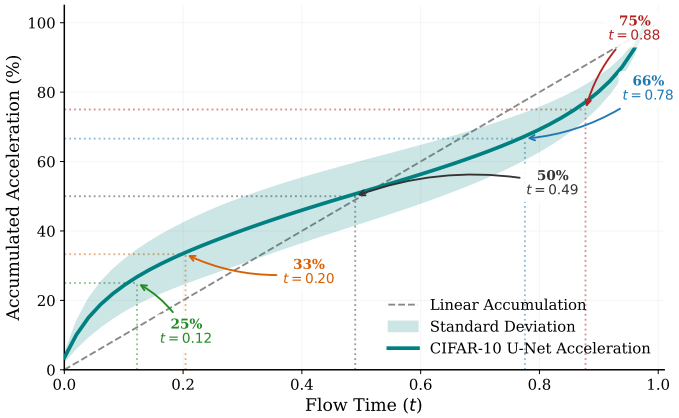
To demonstrate the scalability of ***CBS***, we evaluate its performance as the generative timeline is partitioned across an increasing number of specialized networks. Tab. 6 presents the synthesis quality when using $N \in \{1, 2, 3, 4\}$ models. Rather than relying on heuristic splits that become exponentially harder to tune for a larger number of networks, our complexity-based metric applies unchanged to any N, since the equidistribution condition is exact for any finite N (Sec. 2.2). As shown, increasing the number of networks consistently improves both FID and Inception Score. This indicates that progressively relieving localized capacity bottlenecks through finer temporal partitioning yields reliable generative gains, at a total training cost that grows linearly with N. While we use $N = 3$ as our default configuration to balance synthesis quality with total training overhead, the continued scaling up to $N = 4$ highlights the robustness and generality of our splitting criterion.



(a) SiT (ImageNet-256)



(b) JiT (ImageNet-64)



(c) UNet (CIFAR-10)

Figure 2: **Cumulative path acceleration across datasets and models.** We visualize the cumulative second-order time derivatives (path acceleration) over the generative timeline $t \in [0, 1]$ for SiT, JiT, and UNet.

Table 7: **Comparison of Monitor Functions.** Both monitor functions yield splits that outperform uniform splitting, with improved scores for the path acceleration based function.

Model	FID↓	IS↑	Precision↑	Recall↑
SiT-B/2 (Dirichlet)	31.25	49.95	0.54	0.64
SiT-B/2 (Acceleration)	**30.51**	**52.20**	**0.55**	**0.65**
JiT-B/4 (Dirichlet)	15.40	**29.90**	0.58	0.59
JiT-B/4 (Acceleration)	**15.02**	29.86	**0.59**	**0.61**

E Comparison of Monitor Functions

We compare here the accuracy achieved by the two monitor functions suggested in Sec. 3: the spatial Dirichlet energy and the temporal path acceleration. Tab. 7 presents the performance of both SiT-B/2 and JiT-B/4 when partitioned using each monitor function. While both monitors outperform uniform splitting, the path acceleration-based monitor obtains better FID scores. For this reason, we use it as the default monitor in the rest of our tests. We attribute its greater success to the fact that it measures the final sampling accuracy more directly than the Dirichlet energy, which focuses on the flow field without accounting for the actual sampling process.

F Training Algorithm

Alg. 1 summarizes one training step of *CBS*: a sub-network index is drawn, and the timestep is sampled only from that network's interval. In our experiments, each sub-network is trained in a separate run for the full schedule, which corresponds to fixing i throughout a run.

Algorithm 1 CBS Training

```
# models: N networks, bounds: N+ 1 splits
# x_1: data batch

i = sample_model_idx(N)
t = sample_t(bounds[i], bounds[i+ 1])

x_0 = randn_like(x_1)
x_t = t * x_1 + (1 - t) * x_0

# Predict velocity and compute loss
pred_v = models[i](x_t, t)
loss = metric(pred_v - (x_1 - x_0))
```

G Experimental Setup Details

To rigorously evaluate *CBS*, we design our empirical study across three distinct generative environments, each posing unique spatial and spectral challenges.

High-Fidelity Latent Synthesis (ImageNet-256). Our primary testbed is the ImageNet dataset at 256×256 resolution, which is a standard benchmark for complex, conditional image generation containing 1000 distinct classes. Modeling flows at this resolution is computationally prohibitive, hence we operate in the latent space of a pre-trained autoencoder. We utilize the Scalable Interpolant Transformer (SiT) (Ma et al., 2024) as our baseline flow model architecture. SiT operates on latent patches and scales predictably, allowing us to cleanly evaluate our network ensemble against monolithic baselines of varying capacities (SiT-S, SiT-B, and SiT-XL).

Pixel-Space Synthesis (ImageNet-64). Operating in latent space often smooths out certain high-frequency features. To assess how well our method handles raw, uncompressed spatial gradients,

we evaluate it also over pixel-space generation on ImageNet at 64×64 resolution. We use the Just Image Transformer (JiT) (Li & He, 2025) as the flow model.

Unconditional Generation (CIFAR-10). Finally, we evaluate our method on the CIFAR-10 dataset (32×32 resolution) in a purely unconditional setting. For this task, we swap the transformer backbone for a standard convolutional UNet architecture (Ronneberger et al., 2015), demonstrating that the benefits of complexity-based partitioning are architecture-agnostic.

Evaluation Metrics. To quantitatively assess the performance of *CBS* against the baselines, we employ a comprehensive suite of standard generative metrics. Our primary metric is the Fréchet Inception Distance (FID) (Heusel et al., 2017), which provides a holistic measure of both image fidelity and distributional diversity. We also report the Inception Score (IS) (Salimans et al., 2016) as a secondary measure for class distinguishability and intra-class diversity. Finally, to disentangle the trade-off between synthesis quality and mode coverage, we report Precision and Recall (Kynkäänniemi et al., 2019).

H Implementation Details

Hardware and Compute Resources. All experiments were conducted on a high-performance computing cluster equipped with NVIDIA H200 GPUs. To manage multi-GPU synchronization and efficiently scale our training workloads across the different architectures (SiT, JiT, and UNet), we utilized PyTorch DistributedDataParallel (DDP). Our underlying software environment was built on standard PyTorch and CUDA releases.

Hyperparameters and Configuration. The complete set of training, optimization, and sampling hyperparameters used across all experimental baselines and our specialized sub-networks is detailed in Table 8. To ensure a rigorous and fair comparison, all models within a specific generative domain were trained and evaluated using these identical base configurations unless explicitly stated otherwise.

Table 8: Training and sampling hyperparameters for the baseline architectures evaluated in our experimental setup.

Parameter	**SiT (ImageNet-256)**	**JiT (ImageNet-64)**	**UNet (CIFAR-10)**
Architecture & Data			
Resolution	256×256	64×64	32×32
Space	Latent (VAE)	Pixel	Pixel
Patch Size	2	4	N/A
Batch Size	256	1024	128
Training & Optimization			
Duration	400K Iterations	600 Epochs	400K Iterations
Optimizer	AdamW	AdamW	AdamW
Learning Rate	1×10^{-4}	1×10^{-4}	2×10^{-4}
LR Schedule	Constant	Linear Warmup	Constant
Weight Decay	0.0	0.0	0.0
EMA Rate	0.9999	0.9999	0.9999
Sampling			
Method	Euler	Euler	Euler
Sampling Steps	250	250	250
CFG Scale	1.5	2.9	N/A