
Mode Forcing: A Unifying and Predictive Theory of Generative Modeling

Alexi Gladstone¹, Yilun Du², Heng Ji¹
¹UIUC ²Harvard

Older draft. This is an older draft of the Mode Forcing paper, and some portions are out of date or being updated.

Abstract

Generative modeling has made tremendous progress in the last decade, going from barely generating digits to generating realistic text, video, and audio sequences, driven by a handful of different approaches. In this work, we introduce **Mode Forcing**, a theory¹ that both unifies existing generative modeling approaches into a simple idea, and is predictive of what generative modeling approaches will work in the future.² The core hypotheses behind Mode Forcing are threefold. First, there are two families of generative models, contrastive and reconstructive, and reconstructive models are the family that has scaled best. Second, naively using reconstruction losses for generative modeling results in blurring modes, as the reconstruction loss minimizer is the mean of the modes. Third, all modern scalable reconstructive models design the *training objective* such that the loss minimizer does not blur modes, primarily by breaking the generation process into nearly unimodal³ pieces. More formally, most existing scalable reconstructive generative models factor a multimodal target distribution into a chain of conditionals that are as close to unimodal as the factorization can make them during training, so as to avoid blurring modes; they differ only in the factorization axis and the per-conditional training parametrization. Mode Forcing thus explains why autoregressive and diffusion models scale, while models such as GANs and VAEs have struggled with scaling. Ultimately, we hope Mode Forcing serves as both a lens for understanding why today’s generative models work and a compass for designing tomorrow’s.

Modern Generative Modeling is the Art of Designing a Reconstructive Objective Whose Loss Minimizer Captures Modes Instead of Averaging Them

Correspondence to Alexi Gladstone: ✉ alexigladstone@gmail.com. Work done while supported as a Flapping Airplanes Fellow.

¹We present Mode Forcing both in plain language, intuitive figures, and in full mathematical detail (Section 4.2, Appendix B), so the idea is easy to grasp without giving up rigor.

²Concurrent work using Mode Forcing to come up with new approaches demonstrates that Mode Forcing is predictive [1, 2], we discuss this more in Section 5.1.

³For diffusion and flow, “unimodal” has its standard meaning: the conditional is single-peaked. For autoregressive models predicting distributions, the next-element distribution can itself be multimodal, but it is

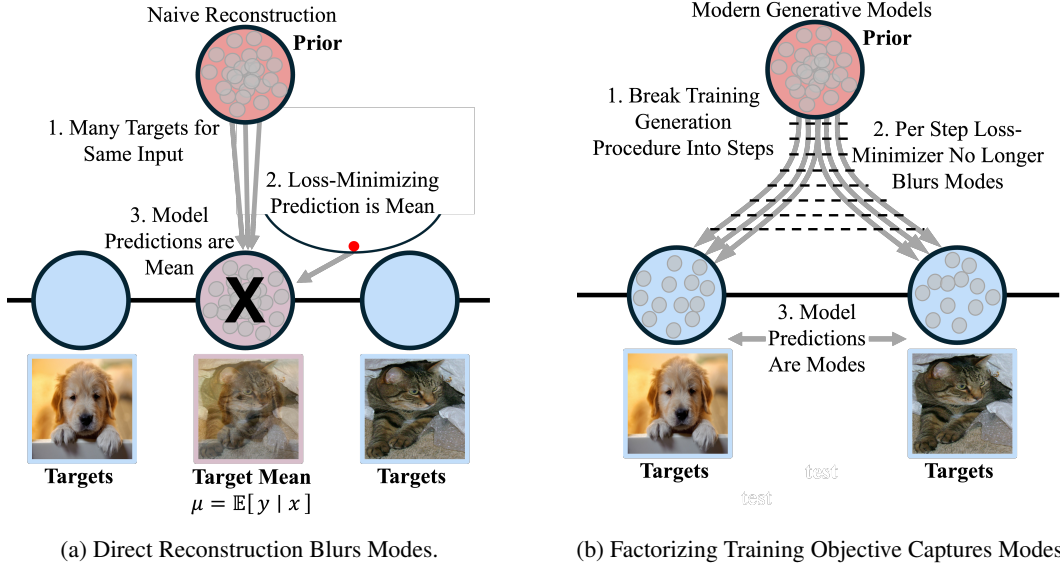


Figure 1: **Reconstruction Blurs Modes; Modern Models Factorize Training to Preserve Them.** Reconstructive generative models train by randomly coupling each source sample to a target sample and minimizing a reconstruction loss between the model’s prediction and the target. Because the coupling is random, a single source point is paired with many different targets across the dataset, so the best prediction that can be made, or the loss minimizer at that point, is the *mean* of those targets rather than any one of them (1a). For a toy 1D distribution, this collapses multiple target modes to their centroid, and on natural images, the centroid is a blurred image. However, when the training objective is instead *factored* so that each piece the model predicts is nearly unimodal (1b), the per-piece loss minimizer lands on individual modes rather than their average, and the modes are captured.

1 Introduction

Why can’t generative models simply train to generate an entire sample in one forward pass? Why do the methods that scale break generation into tokens, timesteps, masks, or trajectories? Autoregressive models are trained to predict one next token, and are then unrolled recurrently for entire sequences at inference. Diffusion models are trained to predict one denoising step, and are iterated recurrently from pure noise to data at inference. These procedures look unnecessarily complex: if the goal is to generate data, *why not train models to generate entire sequences/samples directly, in a single forward pass*? In this paper, we argue that this indirectness is not incidental, but the *central reason* modern generative models work. Direct reconstruction fails because when many outputs are valid, the loss minimizer is their average: a blurred prediction that lands between modes rather than on any real sample. Autoregression and diffusion avoid this failure in the same way: they factor generation into smaller conditional problems, next tokens for autoregressive language models or slightly-less-noisy images for image diffusion models, where each prediction is constrained enough that reconstruction no longer blurs across modes. We call this shared principle of intentionally designing a reconstructive objective to avoid blurring modes **Mode Forcing**.

This principle runs through much of modern generative modeling: autoregressive models [3, 4], diffusion models [5–7], single-step models such as Consistency and Mean-Flow [8, 9], and reconstructive Energy-Based Models [10, 11] each turn out to be a different way of designing the reconstructive objective so that no single prediction is forced to average across modes. The same lens explains observations that otherwise look unrelated: why reconstructive models cannot generally be trained end-to-end, why discrete data has historically been easier to model than continuous data, and why the first sampling step of a diffusion model can be replaced with a precomputed mean at essentially no

uniquely determined by the context (i.e., a single correct distribution over the next token), which is what keeps reconstruction losses from blurring modes.

cost, which we verify (FID 22.56 $\rightarrow$ 22.5). Most importantly, Mode Forcing has already produced new methods, both designed from the theory: Explorative Modeling, which opens a third pretraining axis beyond parameters and data and is able to train reconstructive generative models end-to-end [1], and Jumpy generative models [2]. By contrast, existing accounts tend to do one or the other: some unify many models but cannot say which will scale, while others predict well within a single family but do not unify across them (Section 6). Overall, we built Mode Forcing to be unifying enough to cover the generative models we already have, predictive enough to invent the ones we do not yet have, and simple enough to hold in one’s head.

2 The Untold Story of Generative Modeling

*This section traces the evolution of generative modeling from 2011 to 2026, focusing on how goals have changed over time and which model families became dominant in each era. The pattern to watch is that each era’s goals quietly picked its winners; explaining why they won is the job of the rest of the paper.*⁴

The Beginning of Generative Modeling Before generative modeling was about generation, it was primarily about *representation learning* [12–14]. In the early 2010s, training deep supervised networks from scratch was difficult, and one of the central insights of this era was that a model capable of generating data must have learned meaningful latent structure useful for downstream discriminative tasks. One prominent approach was generative pretraining: train an unsupervised generative model (typically Deep Belief Networks [15] or autoencoders [16, 17]) to capture the data distribution, then fine-tune the learned representations with supervised labels. Generating high-quality samples was not yet a goal; the generative objective was simply a way to learn better features.

This generative modeling approach diminished in popularity around 2012–2014, when advances in supervised learning, such as AlexNet [18], made end-to-end training feasible without generative pretraining. With the representation learning motivation removed, the field of generative modeling needed a new reason to care about generative models.

The Advent of Good Samples The new reason came in the form of *sample quality*. Generative Adversarial Networks (GANs) [19] and Variational Autoencoders (VAEs) [20] shifted the field’s focus from learning useful representations to producing a couple of realistic samples. GANs in particular became the powerhouse of high-fidelity sample generation, producing increasingly photorealistic images [21, 22] that made generating samples the focus and hence the metric of progress, as opposed to just representation learning.⁵

However, neither framework scaled gracefully. GANs relied on contrastive losses between a generator and discriminator, which introduced well-known training instabilities that often worsen with scale (see Section 3.2). VAEs, while more stable to train, suffered from the prior hole problem (the mismatch between the prior and the aggregate posterior; see Section 8), which degraded sample quality as model capacity and data complexity grew. Both frameworks excelled at producing individual compelling samples, but struggled to scale or reliably cover the full data distribution, motivating the field’s next shift toward mode coverage.

The Desire for Mode Coverage Whereas previous eras treated generation as a proxy (first for representation learning, then for demonstrating realism on a few samples), generative models now directly powered real applications: language modeling [4], text-to-image generation [23], and world models for planning and control [24]. The field correspondingly measured progress with metrics that penalize dropping modes: perplexity, long the standard for language models [4], and more recently FID [25] and Inception Score [26] for image generation.

⁴This is one reasonable perspective on the history; we acknowledge that others may have different interpretations, and it is simplified in places for conciseness.

⁵Autoregressive models were already dominant in language modeling throughout this period, for reasons we discuss later.

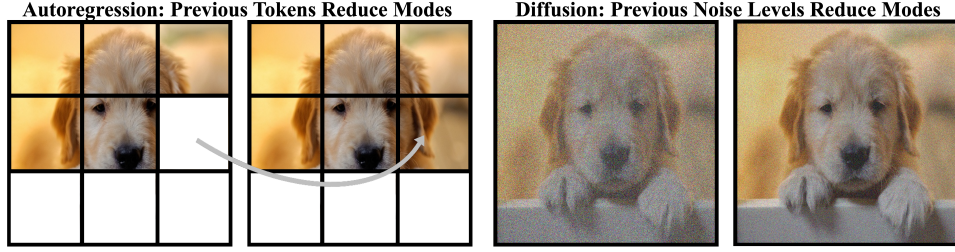


Figure 2: **Conditioning During Training Reduces the Number of Modes.** Autoregressive models (left) and Diffusion/Flow (right) both factorize generation so each step conditions on previously generated information, making the per-step target closer to unimodal. For autoregressive models, prior patches constrain the next patch; for diffusion [6], the previous noisier image constrains the slightly less denoised prediction. In both cases, the model is only ever trained to predict a single step at a time, so per-step reconstruction is a less multimodal target and thus models no longer blurs modes.

Consequently, over time, Autoregressive models [3] and Diffusion models⁶ [5, 6] have emerged as the most common approaches. Notably, both approaches are *reconstructive* (see Section 3.3), training the model to reconstruct data from partially-observed inputs, which directly penalizes dropping modes. As a consequence, however, both approaches decompose generation into a sequence of smaller steps that *condition on ground-truth data during training* but on the *model’s own outputs at inference*, leading to *exposure bias*⁷ (Section 7.1).

Read end to end, the history has a shape: the field’s goals kept changing, and each time they did, a different family of models took over. That is the pattern this paper sets out to explain. Explaining it requires ideas the history did not yet need: what exactly a “mode” is, why reconstruction losses tend to blur modes together, and why only one family of models has scaled. The next section builds these ideas, splitting generative models into two families and isolating the difficulty at the center of the paper; once Mode Forcing is in hand, we return to this history and explain why each era favored the family it did (Section 5.8).

3 Mode Forcing Background

The history showed which family of models won each era; this section begins to explain why, setting up the rest of the paper in three steps. We first explain what makes generative modeling hard (Section 3.1). We then classify generative models into two groups: contrastive (Section 3.2) and reconstructive (Section 3.3) models. The value of this grouping is that all approaches within a family share the same benefits and limitations, which both unifies our understanding of existing generative models and gives us a framework for classifying new ones. Finally, we discuss the coupling problem (Section 3.4): the difficulty, specific to reconstructive models, of deciding which input should map to which target.

3.1 What Makes Generative Modeling Hard

Generative modeling asks for something supervised learning does not. In classification or regression each input has essentially one correct output, so a single deterministic mapping suffices; generation has no such mapping to learn, because its target is a whole distribution rather than one answer. A request for “a dog” has no single right output, since any breed, pose, or background will do. These valid outputs are the *modes* of the distribution, the separate peaks where probability mass concentrates, and natural data has a great many of them, spread across a thin, intricate manifold [28, 29]. This is what makes generation hard: a model that regresses straight from input to output must commit to one prediction even though many are valid, so under a reconstruction loss its best output is their average,

⁶We broadly use the terms Diffusion and Flow interchangeably, unless otherwise noted, as they can be viewed as the same [27].

⁷We broadly define exposure bias in this paper as a mismatch between what the model sees during training and inference.

a blurred blend that lands between the modes rather than on any real sample (Figure 1). Avoiding this collapse while still covering every mode is the *root of nearly every difficulty* in generative modeling, and it is why we organize generative models by how they treat modes: whether they contrast them or are directly trained to reconstruct them.

3.2 Contrastive Models

We call a generative model *contrastive* if its training objective requires distinguishing real samples from contrast samples, such as model generations, noise, or other distractors, rather than being trained to explicitly reproduce any training samples (Definition B.1). This family includes GANs [19, 21, 22, 30], Contrastive-Divergence-trained Energy-Based Models [31, 32], and InfoNCE-based models [33] such as Drifting Models [34]. As discussed in Section 2, GANs in particular dominated visual generation from 2014 through roughly 2021, with notable applications in image-to-image translation [35, 36] and super-resolution [37].

Contrastive models share a characteristic profile of strengths and weaknesses, all stemming from the fact that they are trained to distinguish real from contrast samples rather than reconstruct any particular example. On the strength side, the absence of reconstruction signal means contrastive models avoid mode blurring (Figure 1) and are not subject to exposure bias (Section 7.1), giving them sharp, high-fidelity samples on the modes they do cover [21]. The weaknesses stem from the same property: because the objective only requires distinguishing real from contrast, its loss minimizer does not need to place mass on every mode of the data distribution, only enough to fool the contrast. As a consequence, contrastive models often have poor mode coverage [38]. They have also not scaled as gracefully as reconstructive models so far; one likely reason is that distinguishing real from contrast in high dimensions appears to require a number of negative samples that grows quickly with dimension [11, 39] (we discuss this further in Section B.1). Training stability is a recurring difficulty as well: in GANs, due to the minimax dynamics between generator and discriminator, and in CD-EBMs, due to biased gradient estimates and poor MCMC mixing [11, 40].

This profile (sharp samples on a few modes, fast inference, poor coverage, hard to scale) aligned almost perfectly with the priorities of the GAN era (Section 2): sample quality was the dominant metric and mode coverage was not yet a serious concern. Once the field shifted toward metrics that penalized mode dropping, the same profile became a liability, and reconstructive models, which have the inverse trade-off, began to displace contrastive ones.

3.3 Reconstructive Models

We call a generative model *reconstructive* if its training objective requires the model to directly predict individual training samples (or parts of them) from a conditioning input, such as a partial observation, a corrupted version, or a coupled noise sample (Definition B.2). This family includes Autoregressive models [3, 4], Diffusion and Flow Matching models [5–7], Variational Autoencoders [20], Mean-Flow models [9], and reconstructively-trained generative Energy-Based Models [11]. As discussed in Section 2, reconstructive models dominate generative modeling today, powering modern large language models [4], text-to-image generation [23], video generation models [41], and world models for planning and control [24].

Reconstructive models have the mirror-image profile. Because they are trained to reproduce real samples rather than tell them apart from contrast samples, the loss minimizer is forced to answer for every mode, which pushes them toward strong coverage. Training is a regression problem with no negative samples and no adversarial game, so it scales to high dimensions and stays stable. The cost appears when the target is multimodal: a naive reconstruction loss averages the modes together, since its L_2 minimizer is the conditional mean. This mean-blurring is precisely the problem Mode Forcing addresses and that we seek to explain: all modern scalable reconstructive models avoid this blurring by factorizing the training objective into approximately unimodal sub-problems, so that per-piece reconstruction no longer averages across modes (Figure 1). However, this factorization introduces its own costs: often inference becomes multi-step and conditions on the model’s own outputs, leading to exposure bias and slower generation. This explains why popular generative models such as Autoregressive and Diffusion models are trained to predict a single step (Figure 2) yet run as Recurrent Neural Networks at inference (Figure 3). Single-step models, such as Consistency and Mean-Flow models [8, 9], avoid this recurrence but suffer from a different type of exposure bias:

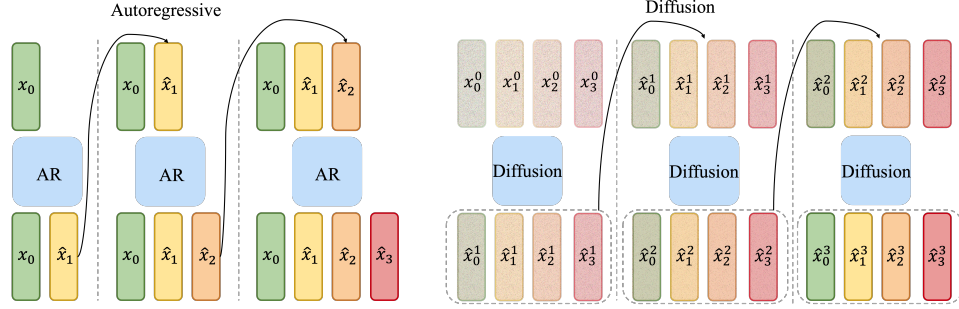


Figure 3: **Autoregressive and Diffusion Models are Recurrent at Inference.** Autoregressive models (left) and Diffusion/Flow (right) are both inferred recurrently, using their own previously generated predictions as inputs for future steps. This contrasts with how they were trained, which was to predict a single step (Figure 2), and therefore results in *exposure bias*. This mismatch cannot simply be trained away: training these models recurrently would blur modes (Section 5.4).

they train primarily on factorized sub-trajectories of the noise-to-data path, yet at inference they often jump the full trajectory in a single step.

3.4 The Coupling Problem

Training models via reconstruction works by training to map from an input to a target (i.e., a partial sequence, to the partial sequence with another token; or a noised sequence, to a slightly less noised sequence). Because of this, reconstructive training needs a *coupling*. A reconstruction loss is computed on input-target pairs: the model sees an input and learns to reproduce its target. The target is some version of the data we want to generate, but the input cannot be, since at inference we have no data to start from. So, the input has to be something we can produce on our own, like a noise vector, a corrupted sample, or a partial observation, and the coupling is what pairs each such input with a target. The challenge with reconstructive generative modeling is that the right coupling is not known in advance. If it were known, generation would collapse into supervised regression, and there would be no generative modeling problem left to solve.⁸

Lacking that knowledge, the standard choice is to couple at random: draw a noise sample and attach it to a randomly chosen target.⁹ This is convenient, but it is precisely what creates mode blurring: because the pairing is arbitrary, one source is matched with many different targets across the dataset, so the reconstruction minimizer at that source is their average (Section 4.1). One might hope to dodge the problem with a smarter coupling, such as an optimal-transport pairing that sends nearby sources to nearby targets. This does not resolve the issue in practice. Exact optimal transport is intractable at modern dataset scale (its cost grows cubically in the number of samples), and minibatch approximations [42] solve transport only within each batch, so the coupling they recover is a biased proxy for the true global one [43, 44], and a batch is only a vanishing fraction of the data. Its gains are correspondingly narrow: they appear on small or downsampled data and mainly when taking few sampling steps, shrinking or even reversing as the step budget grows [45]; at scale, high-resolution flow models instead train with independent coupling [46]. It can also hurt outright: in conditional generation the matching ignores the conditioning and skews the training prior away from the one used at sampling [47]. Rectified flow [48] learns a straighter coupling by repeatedly regenerating targets from the model itself, but the coupling it finds is then capped by the quality of the existing trained model. Hard-coding a coupling can also stop the model from discovering useful structure of its own.

⁸If the coupling were known, each input would be tied to a single target rather than many, so the conditional we regress is a point mass whose mean is that one target, meaning there is nothing left to average over, and thus no mode blurring. Knowing the coupling is the same as being handed the latent variable conditioning that distinguishes which mode each input belongs to.

⁹Autoregressive models sidestep this random coupling by making each input for couplings a real partial observation—usually the preceding elements in a sequence—paired with the sequence’s true continuation.

Mode Forcing is the claim that every scalable generative model resolves this same coupling issue in the same way, which is to design the reconstructive training objective such that models no longer blur modes.

4 Mode Forcing

We now state Mode Forcing and build it up from first principles. We start with the intuition (Section 4.1), using a toy example to show visually why naive reconstruction blurs modes and how factorization fixes it. We then make the argument precise (Section 4.2): the loss minimizer of a reconstruction objective is a blurred mean, and every modern model is a different way of factoring the target so that this minimizer stops averaging across modes. We then ground the abstract picture in named models (Section 4.3), and later revisit the history from Section 2 through this lens (Section 5.8). We address common counterarguments or counterexamples to Mode Forcing in Section 8.

4.1 Mode Forcing Intuition

The idea behind Mode Forcing is simple to state: modern scalable generative models are reconstructive (Section 3.3) and function by factorizing the training objective into nearly unimodal pieces, so that a reconstruction loss can be used without blurring modes. We build the intuition for Mode Forcing here with a toy example, then make it precise in Section 4.2.

The problem: reconstruction with a random coupling blurs modes. Suppose we want to turn random noise into samples from a target distribution, say the three piles of data or a pool of dog images (Figure 4). The most direct approach is to train a network to map each noise sample straight to a data sample under a reconstruction loss such as L_2 . This fails in a specific, predictable way. Because we pair noise to data *at random* (we have no a priori reason to send a given noise vector to one target rather than another; see Section 3.4), each noise input is paired with many different targets over the course of training. The best a squared-error predictor can do is output the *average* of those targets (Figure 1). For the three piles, that average is the empty centroid between them; for dog images, it is a blurry mean dog. The model’s prediction lands where there is no data, resulting in poor performance.

Why not just avoid reconstruction? One reaction is to drop reconstruction losses entirely, as contrastive models do; with no targets to average, they generally don’t blur modes (Section 3.2). But contrastive training has not scaled well thus far: telling real from contrast samples in high dimensions appears to require a number of negatives that grows quickly with dimension, a manifestation of the curse of dimensionality [39]. Thus, reconstructive models are the only family that has scaled gracefully so far, and therefore the mode-blurring problem must be addressed. Much of Mode Forcing is the observation that the modern generative models (e.g., Autoregression, Diffusion [6], Consistency [8], etc.) are each a different way of addressing this problem.

The fix: factor the target into nearly-unimodal pieces. The blur comes from a target distribution with many modes. If instead each piece the model predicts has essentially *one* mode, the average and the mode coincide, and reconstruction succeeds. There are two complementary ways to get there, and together they cover all existing reconstructive models known to date:

- **Train with a factorized reconstructive generation objective.** Diffusion/Flow split the noise-to-data map into many small denoising steps; autoregressive models split a sample into a sequence of next-element predictions. In both, conditioning on what came before (a slightly-less-noisy image, the previous tokens) reduces ambiguity about future pieces, so each per-step target is nearly unimodal (Figure 2).
- **Train with an explorative reconstructive generation objective.** Rather than asking a single prediction to land close to every target at once (which blurs them together), Explorative Modeling [1] lets the model make several guesses and updates only the closest one. Because only the best guess is reinforced, the guesses are free to spread out and settle on different modes, together covering them all (bottom two rows of Figure 4). The same idea works in reverse (Reverse XM), keeping a single guess but matching it to the nearest of several targets; either direction reinforces only the best-matching pair, though the reverse direction

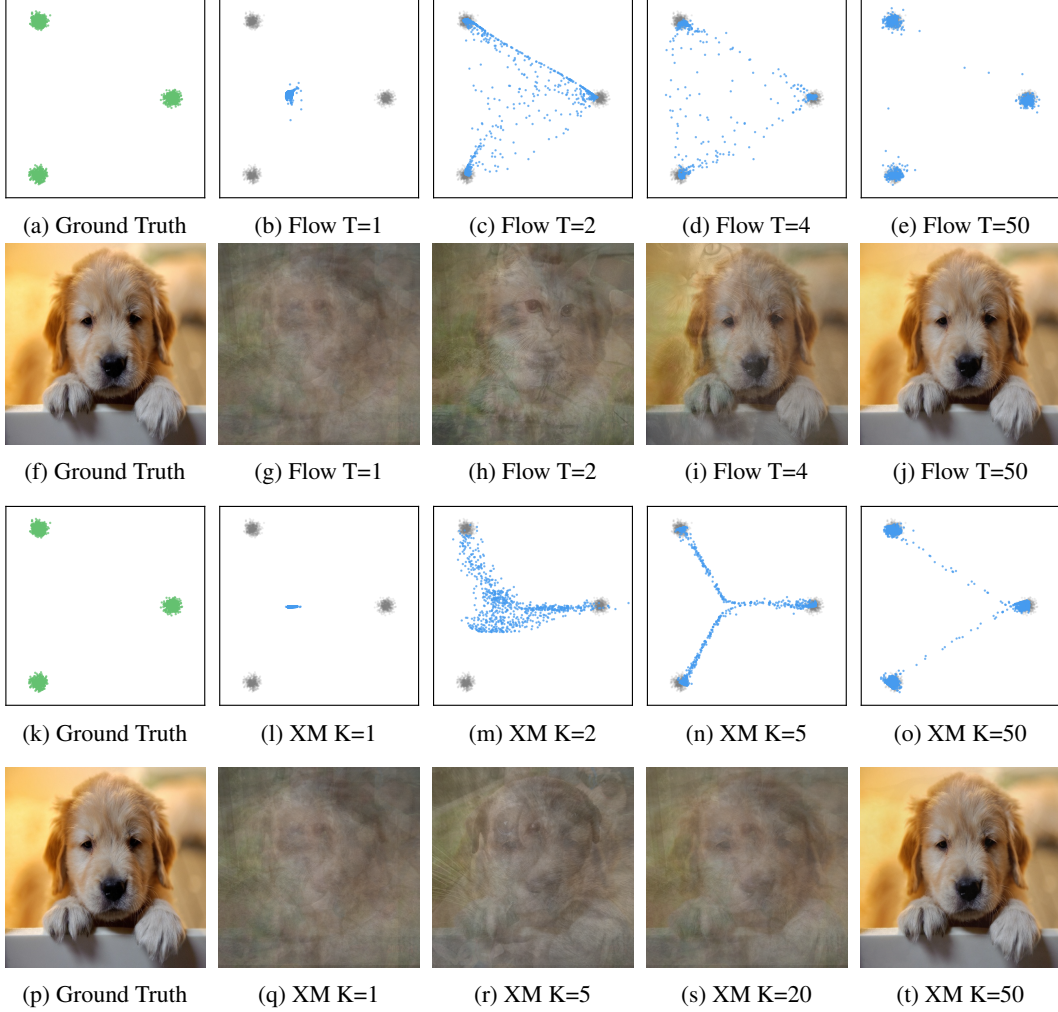


Figure 4: Factoring or Exploring the Objective Improves Coverage and Reduces Blurring. To demonstrate the effect of Mode Forcing’s two routes on models’ ability to handle multimodal distributions, we use one method for each and progressively increase it. The first method (top row) factors the objective, training models via Diffusion/Flow [6, 7] but using only T timesteps. The second, Explorative Modeling (XM) [1], explores the objective instead, sampling K predictions during training and only backpropagating the minimum-loss one. Both routes progressively bring models from blurring modes to capturing modes, reinforcing Mode Forcing’s main hypothesis; because they supply the same thing, they can also substitute for one another [1].

errs toward precision rather than coverage.¹⁰ Either way, the procedure can be intuitively seen as searching for which mode to commit to instead of averaging them, or equivalently as fixing the random coupling that causes blurring in the first place.

To directly demonstrate the effects of these approaches, we run controlled experiments (Figure 4) in which we ablate the number of ‘pieces’ in the training objective, controlling how many modes get blurred. We use two approaches, one for each way of handling multimodality. The first, corresponding to the factorized objective, trains Flow with T discrete timesteps.¹¹ Here $T = 1$ corresponds to end-to-end direct regression in a single step, whereas $T \rightarrow \infty$ recovers standard flow matching [7]. Elegantly, this ablation recovers Jumpy Generative Models [2], concurrent work derived from Mode

¹⁰In principle, the forward and reverse variants can also be combined [1].

¹¹Flow is not normally used with discrete timesteps; here this amounts to sampling only T distinct values of t during training. As $T \rightarrow \infty$ this becomes equivalent to normal Flow.

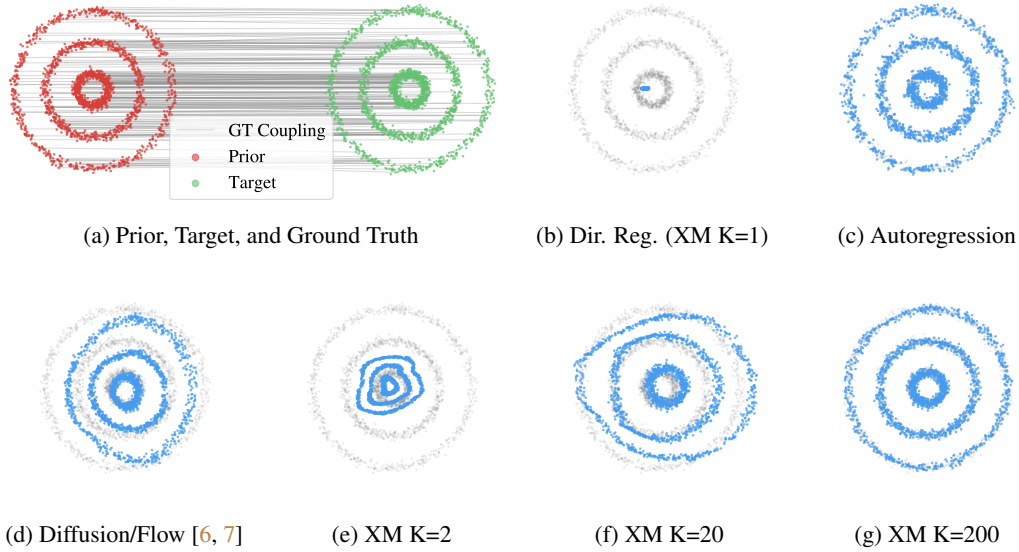


Figure 5: **Mode Blurring Occurs When Prior and Target Are Same.** In this experiment, the prior distribution and target distribution are the *exact same* (both are dartboards). Despite this setup, almost paradoxically, models performing direct end-to-end prediction of the target from the prior regress the mean. Mode Forcing predicts exactly this: because the coupling from prior to target is sampled at random (Section 3.4), each prior point is paired with many different targets across training, so the loss minimizer at that point is the conditional mean of all the modes rather than any single one (Figure 1). Autoregressive models and Diffusion/Flow both avoid this pitfall by factorizing the training process into pieces that are nearly unimodal (Figure 2). Similarly, Explorative Modeling (XM) [1] loops over K candidate predictions and only backpropagates the lowest-loss one, reshaping the loss minimizer so it snaps to a single mode instead of their average. Dir. Reg. stands for Direct Regression.

Forcing that sets the number of jumps to match how multimodal the data is. The second approach, corresponding to the explorative objective, is Explorative Modeling (XM) [1]: we sample K candidate predictions during training and only backpropagate the lowest-loss one, so that $K = 1$ is again plain end-to-end regression and larger K lets the candidates spread across the modes.

These results demonstrate that both approaches for handling multimodality reach the same place as the number of ‘pieces’ increases: the effective per-prediction target becomes nearly unimodal, which allows the model to successfully generate realistic samples (Figure 4).¹² Together, the two ablations are direct evidence for Mode Forcing’s central claim: a reconstruction loss generates well precisely when the objective is designed to be nearly unimodal.

Blurring is about coupling, not prior and target distribution mismatch. It is tempting to blame mode blurring on the gap between the noise distribution and the data distribution. Figure 5 shows it is not. Even when the source and target are the *exact same* distribution (a dartboard mapped to itself), one-shot regression still collapses to the mean. The cause is the random coupling (Section 3.4): each source point is paired with many targets, so its loss-minimizing prediction is their average. The same factorization that helps above fixes it here too.

4.2 Mode Forcing Theory

We now state the central claim of Mode Forcing precisely. Let $p^*(y | c)$ be the data distribution we want to sample from, with optional conditioning c (a class label, a caption, or nothing at all). A reconstructive model fits a predictor f under a loss of the form $\mathbb{E}_{c, y \sim p^*(\cdot | c)} [J(y, f(c))]$, where J measures reconstruction error. When J is squared error (or any Bregman divergence), the minimizer

¹²Although we present the two approaches as distinct, both are simply ways of handling multimodality, and they can be combined, which often improves performance [1].

is the conditional mean:

$$f^*(c) = \arg \min_f \mathbb{E}_{c, y \sim p^*(\cdot | c)} [\|f(c) - y\|^2] = \mathbb{E}[y | c].$$

This single equation is the entire problem. The best possible predictor outputs a single *point*, the average of every target consistent with c , and so collapses all the modes of $p^*(\cdot | c)$ onto their centroid (Figure 5). Conditioning does not rescue us: for $c = \text{"dog"}$, $\mathbb{E}[y | c]$ is the average over all dog images, a blurry mean dog rather than any real one.

Two ways to force modes. Every scalable model escapes this by reshaping the problem so the minimizer is no longer a blurred mean. There are two ways to do it, the same two objectives described in Section 4.1.

1. **Factor the objective.** Corrupt the target and learn to reconstruct it conditioned on a partial view of itself, so each piece the model predicts is far less multimodal than the whole. Two corruptions are used. *Denoising* models (diffusion, flow, mean-flow, consistency) condition on a noised version of the target,

$$p^*(y | c) = \int p(y_0) \prod_{t=1}^T p^*(y_t | y_{t-1}, c) dy_{0:T-1},$$

and regress each step; conditioning on the slightly noisier state brings each per-step target as close to unimodal as the factorization can make it, so per-step mean prediction barely blurs. *Masking* models (autoregressive models, and masked discrete diffusion [49, 50]) instead condition on a masked version, reconstructing hidden elements from revealed ones,

$$p^*(y | c) = \prod_i p^*(y_i | y_{<i}, c),$$

which shrinks each piece toward one mode in the same way. Where a piece is still multimodal, these models express it directly instead of averaging it, with a softmax over a discrete vocabulary or a mixture / small diffusion head over a continuous one. The finer the factorization, the less of this is needed: denoising takes near-infinitesimal steps and can regress a bare mean, while a model that commits a whole element at once leans on the softmax to carry the multimodality that remains.

2. **Explore the objective** (winner-take-all selection; Explorative Modeling, XM [1]). Keep a single shot, but let the model emit K stochastic candidates and reinforce only the one closest to the target,

$$\mathbb{E}_{c, y \sim p^*(\cdot | c)} \mathbb{E}_{\hat{y}_1, \dots, \hat{y}_K \sim G_\theta(\cdot | c)} \left[\min_{k \in \{1, \dots, K\}} \|\hat{y}_k - y\|^2 \right],$$

where the generator G_θ produces each candidate $\hat{y}_k = f(c, z_k)$ from its own random draw z_k (this is the Forward variant of XM [1]). With $K = 1$ this is exactly the blurred regression above; with $K > 1$, because only the best guess is reinforced, the candidates are free to spread out and settle on different modes, covering them as K grows, and the model is never penalized for committing to one; the exploration searches for which mode to commit to instead of averaging them. Equivalently, choosing which candidate to reinforce lets the model decide which target each source maps to, rather than accepting the random pairing it was handed: exploring repairs the random coupling that caused the blur in the first place, instead of shortening the chain. Because it never factors the generation, this is the one reconstructive approach that can stay completely end-to-end while capturing multimodal distributions (Section 7.1): a normalizing flow never simulates its own sampling pass during training, and a mixture density network never samples from its own mixture during training either, so neither is end-to-end in the sense we mean (Section 8). The cost is that K must grow with the number of modes, which is why the approach is at present most practical where that number is modest [1].

Both do the same job: *keep any single prediction from being forced to average across modes.*

Generative expressivity: a third bottleneck. There is a clean way to see what these mechanisms buy. Deep learning already recognizes two ceilings on what a model can become. *Parameter expressivity* is the familiar one: a network can only represent functions its size and architecture allow, which is why we scale parameters. *Data expressivity* is the other: a model cannot learn structure its dataset never shows it, which is why we scale data. We argue generative modeling has a third ceiling, *generative expressivity*: the largest number of modes an approach can ever represent, set not by the network or the dataset but by the training objective itself. The blur above is really a failure of this third kind. The best a direct regressor can output is a single point (Figure 5), so its generative expressivity is exactly one mode, and no amount of the other two resources moves that ceiling: with unlimited parameters and unlimited data, the loss minimizer is still the one blurred mean, and a bigger network merely reaches it sooner. Nor is the escape to “do maximum likelihood properly”: squared-error regression already *is* maximum likelihood, under a Gaussian of fixed width, so the likelihood is fit faithfully—just over a family that can hold one mode, whose best fit to multimodal data is the mean [1]. The only way past this bottleneck is to reshape the objective: factoring into pieces multiplies the ceiling, an explicit softmax or mixture output raises it to the size of that distribution, and winner-take-all selection lifts it to at least the number of candidates.¹³ Once the objective is factored finely enough, what separates one model family from another is how much of this ceiling a single prediction inside it is allowed to carry (Appendix B.3). Generative expressivity may also explain why likelihood has long been observed to correlate poorly with sample quality [51]: likelihood measures how well a density is fit, while expressivity sets how many modes that density can hold. Appendix B makes all of this precise, defining a mode count, the generative expressivity each mechanism reaches, and an expected-blurring measure $\mathcal{B}$ with a floor and a per-prediction ledger for each of the two routes.

4.3 How Existing Models are Mode Forced

With these two routes in hand, familiar models fall into place. **Diffusion and flow** are denoising models: each step regresses a velocity or score, and the trajectory turns one multimodal jump into many near-unimodal ones. **Autoregressive models** are masking models: conditioning each token on those already generated shrinks its distribution, and a softmax then expresses whatever multimodality remains, so no averaging happens even though each token is predicted in “one shot.” This is also why autoregression has always felt natural for language but awkward for pixels: a softmax can express a multimodal distribution over a discrete vocabulary exactly, but there is no equally simple way to do so over a continuum. This is why continuous autoregressive image models lean on discretization (VQ) [52], or attach a small diffusion or flow head to each token to model its continuous distribution [53].¹⁴ Masked discrete diffusion sits here as well, despite the “diffusion” name: it corrupts by masking tokens rather than adding noise. **Mean-Flow models** [9], along with the closely related consistency [8] and shortcut [55] models, are subtler denoising models: they keep the trajectory but condition each prediction on where it sits along it and how much of it to cover in one step (by its time, its step size, or both), which keeps each prediction’s target near-unimodal even as the number of inference steps drops as low as 1. What differs is how each stays anchored to the underlying flow field. A consistency model maps any point on the probability-flow trajectory to its clean endpoint, with a self-consistency constraint that forces points on the same trajectory to agree, so the deterministic ODE fixes a single one-step target instead of a blurred average. A shortcut model conditions on the step size and trains so that one large step matches two consecutive half-steps, anchoring large jumps to compositions of small, near-unimodal ones. Mean-Flow predicts the average velocity over a time interval and ties it to the instantaneous flow velocity through an exact identity, so even a single large step is the time-average of the same near-unimodal field that ordinary flow matching regresses. **Jumpy** generative models [2] function by factorizing the generation procedure based on how multimodal the data is. **Explorative Models** [1] function by exploring the loss landscape and only backpropagating on lowest loss samples, allowing guesses to spread out and settle on different modes. Regardless of the details, all modern scalable reconstructive models function in the same way—which is to *force modes by designing a factorized or explorative training objective*.

¹³Shown in Appendix B.5 for a smoothed form of the winner-take-all objective, which differs from the implemented hard min by at most $\log K$.

¹⁴There are some approaches that avoid the VQ or diffusion/flow heads, but they often rely on other approaches such as a mixture of Gaussians [54].

5 What Mode Forcing Predicts Correctly

A theory is only as useful as its predictions. If Mode Forcing merely re-described existing models, it would be a tidy summary and little more. Its real value is that it tells us, *before* running an experiment, where mode blurring will show up and how to remove it. This section collects predictions and explanations that follow directly from Mode Forcing: some have been borne out by concurrent work or simple experiments, and others account for patterns the field already shows, including its own history.

5.1 A New Way to Force Modes (Explorative Modeling)

Mode Forcing says the factorization axis is incidental; what matters is only that each piece the model learns is close to unimodal. This implies the two factorizations everyone uses, the diffusion/flow trajectory and the autoregressive chain, are not the only options. Explorative Modeling (XM) [1] takes a different route entirely: rather than cutting the target into smaller steps, it lets the model propose K candidate samples and trains only on the one closest to the target (winner-take-all). With $K = 1$ this is ordinary regression and blurs; as K grows, the candidates spread out and snap onto individual modes (Figure 5, bottom rows of Figure 4). This is what Mode Forcing would predict: anything that drives the per-piece conditional toward unimodal removes the blur, regardless of how it does so.

5.2 Factorization Should Scale with the Number of Modes

Because blurring comes from multimodal targets, Mode Forcing predicts that the amount of factorization a model needs should grow with how multimodal the data is. Jumpy Generative Models confirm this: the best number of jumps increases with the number of modes in the distribution [2]. The same logic explains a practical puzzle: why image generation tends to need more sampling steps than video or world modeling. When a model is conditioned on the recent past (previous frames, a world state), most of the uncertainty is already resolved, so the per-step conditional is closer to unimodal and fewer pieces are needed. Unconditional or weakly conditioned image generation has many more modes per step, and so demands finer factorization.

5.3 The First Generation Step Is Just the Conditional Mean

A sharper, falsifiable prediction: at the very first step of a diffusion or flow model, the input is pure noise and carries no information about which mode the sample will become. With nothing to disambiguate, the loss minimizer at that step is exactly the mean of the (conditional) data, the “average dog” for the class dog (Section 4.2). Mode Forcing therefore predicts that we can replace the first, most expensive-to-justify network evaluation with the precomputed class mean $\mathbb{E}[y | c]$ and lose essentially nothing in sample quality. This is easy to check: we do this by overriding the first sampling step of a class-conditional ImageNet [56] model with the class mean and run a standard evaluation. On a DiT trained for 1.6M steps, the baseline gives FID 22.56 and replacing the first step with the precomputed per-class training-set mean gives FID 22.5, confirming the prediction. Beyond confirming the theory, this points to a small “free” speedup, and it connects to analytic treatments of diffusion’s early, high-noise steps [57].

5.4 Why We Cannot Generally Train Reconstructive Generative Models End-to-End

The same logic predicts what would happen if we collapsed the entire chain into a single end-to-end network and trained it on a reconstruction loss against ground-truth samples (i.e., training the models in Figure 3 recurrently during training). Once the intermediate steps no longer condition on ground truth, the chain composes into a single deep network from noise to data, and a reconstruction loss against that map has the same minimizer as ordinary direct regression: the conditional mean of the data (Figure 1). The modes blur, because a deeper network with intermediate stops is more parameter-expressive but no more *generatively* expressive than the direct regressor analyzed in Section 4.2: its optimum is still the one-mode conditional mean. This is why end-to-end generative modeling is hard, because even if we had the compute to unroll diffusion or autoregressive models as RNNs during training, doing so would result in blurring modes. Explorative Modeling is the exception: it trains end-to-end under a reconstruction loss, yet avoids this collapse because reinforcing only the best of

K stochastic candidates reshapes the loss minimizer away from the blurred mean (Section 5.1). We return to this as an open problem in Section 7.1.

5.5 Where to Spend Factorization (Noise Schedules)

The same reasoning predicts the shape of a good diffusion noise schedule. Steps near pure noise face the most modes and contribute the most blur, while steps near clean data are nearly unimodal and easy. Mode Forcing therefore predicts that effort should not be spread evenly across noise levels: it should concentrate where the residue is most *reducible* rather than simply largest. At the noisiest end the target is maximally multimodal but the input carries nothing to disambiguate it, so the residue is irreducible and the step is nearly free (Section 5.3); near clean data there is little residue left to remove. The effort is worth spending in between, where conditioning has begun to bite and finer factorization still buys coverage. This connects to the way noise schedules and per-noise-level loss weightings are deliberately shaped rather than left uniform [6, 58], and to the schedule adjustments that richer, higher-dimensional latent spaces have been found to need [59].

5.6 The Mode Collapse Hypothesis

Mode Forcing also suggests a hypothesis worth testing in more detail: modern generative models, even the ones that have achieved strong empirical performance (e.g., autoregressive and diffusion models), may still be somewhat mode collapsed, with some residual mode blurring. Generative expressivity (Section 4.2) suggests why some of this issue with mode collapse should remain, and even *grow with scale*, provided we read it at the level of a single prediction: a fine enough factorization can in theory carry far more modes than any dataset presents, yet what a trained model actually retains falls well short of that, so what binds is how many of them any one prediction inside it is allowed to keep (Appendix B.3). A model’s generative expressivity is set by how it factorizes/explores the target, rather than by its parameter count, so scaling the network alone does not necessarily let it cover more modes if generative expressivity is the bottleneck (i.e., an end-to-end regressor, no matter the size, will only ever be able to capture a single mode, see Figure 1); meanwhile the data we ask it to fit often grows more multimodal as we scale, while how many of those modes a single prediction inside the factored procedure can retain is set by the objective and does not move with scale (we use the same Diffusion/Autoregressive formulation regardless of scale). We therefore expect residual mode collapse to affect models **more** at scale, much as parameter expressivity limits a fixed size network more when trained for longer [60].

The clearest evidence for this hypothesis is the strong reliance of modern models on guidance. Classifier-free guidance [61] pushes samples *away* from what a weaker, more mode-blurred model would produce (as unconditional models have less conditioning, there are more modes, and thus they blur the modes more). Autoguidance [62] makes this point even clearer: it generally achieves better performance than CFG by steering away from a deliberately *worse*, more mode-collapsed version of the same model (i.e., Autoguidance often uses a smaller model trained for less time). This is what Mode Forcing would predict if guidance is really about being pushed away from the conditional mean: a more mode-blurred reference to steer away from helps more (at least, up to a point). It also explains why you cannot get the same effect by simply scaling up the model’s own score or velocity: what matters is the direction guidance pushes, away from that blurred reference, not the magnitude of the prediction itself. Some reward-based fine-tuning of language models may serve a similar function, though it is less clear whether that is the case, as discrete models are often less susceptible to mode blurring (Section 5.7).

5.7 Why Discrete Data Is Easier than Continuous Data

A final pattern Mode Forcing explains: the mode-blurring problem at the heart of this paper is less of an issue when the data is discrete. Over a finite vocabulary, a model can output a normalized distribution directly, a softmax over every possible next element, and the loss minimizer of that output is the full conditional distribution rather than a mean (Section 4.2), so no matter how many continuations are valid, nothing is blurred. A continuum admits no equally cheap normalized output, so continuous data must force modes another way: factor the target into a fine trajectory, as diffusion does, or make the data discrete. Discreteness removes only this one difficulty: pixel-level autoregressive models were discrete yet still lost to diffusion on images, but their problems, sequence length and sampling cost, lie on a different axis from blurring modes.

5.8 The Evolution of Generative Models, Revisited

We can now close the loop on the history from Section 2. Each era’s dominant model was simply the one whose way of handling modes matched what that era measured. When generation was a proxy for representation learning, sample quality, and therefore mode blurring, did not matter, so no factorization was needed. When the field began to prize a few striking samples (the GAN era), contrastive models were ideal: by never reconstructing, they never blur, giving sharp samples, and their poor coverage and shaky scaling simply were not being measured (Section 3.2). Once metrics like perplexity and FID made mode coverage and scale the targets, that profile became a liability, and the field turned to reconstructive models, which cover modes and scale but must defeat mode blurring to do so. Mode Forcing is what they all converged on: design the training objective so that its loss minimizer no longer blurs modes, primarily by factoring the target until each piece is nearly unimodal.

6 Related Work

Unifying frameworks. Several lines of work unify *subsets* of generative models. Score-based stochastic differential equations [63] cast diffusion and flow matching as one continuous-time family [7, 27], building on the classic link between denoising and score matching [64]. These unifications are elegant but partial: they bring diffusion, flow, and score models under one roof while leaving out autoregression, the other family that scaled. Mode Forcing spans both, because it organizes models by what their loss minimizer does to *modes*, not by the form of their dynamics.

Likelihood and divergence as a common foundation. A broader framing treats maximum likelihood, or a variational bound such as the ELBO [20], as the principle behind generative modeling. The difficulty is that almost everything fits: VAEs, normalizing flows, autoregressive models, diffusion, and energy-based models can all be written as maximizing a likelihood or a bound on one, including many approaches that never scaled. A description that fits the failures as well as the successes cannot tell us which is which. Mode Forcing asks a narrower, operational question: does the per-piece loss minimizer average across modes? That question separates the models that scaled from the ones that did not.

Explaining why generative models work. There is also substantial work explaining particular successes after the fact, from analytic studies of the diffusion process [57] to scaling-law analyses of large models [60, 65]. Such accounts tend to be model-specific and descriptive rather than predictive across families. We aim instead for a single, compressive lens that applies to autoregression and diffusion alike and predicts which new approaches should work (Section 5). We expand on this view of a theory as compression in Appendix A.

7 What’s Left for Generative Modeling?

Generative modeling has made enormous progress over the last decade, but several problems remain wide open. Below, we highlight the problems we see as most important, viewed through Mode Forcing.

7.1 Reducing Exposure Bias and Achieving Truly End-to-End Generation

Factorization buys mode coverage, but it has a cost. A model that conditions each step on the ground-truth previous piece during training must, at inference, condition on its *own* previous outputs; small errors then compound across steps. Put another way, these models are trained like autoencoders, each step reconstructing a target from a given input, yet they behave like recurrent networks at inference, feeding each output back in as the next step’s input. This train/inference mismatch is known as *exposure bias* [66, 67], and we see removing it as the central unsolved problem for the field.

We also think the common framing of “end-to-end” generative modeling is incorrect. End-to-end is usually interpreted to mean “generate in a single step,” and many recent methods claim it on that basis [55, 68]. But single-step generation is a statement about *inference speed*, and as compute keeps growing exponentially [69, 70], we argue that inference speed is not the right objective to optimize

for. Instead, we argue that the property that matters is the *absence of a train/inference gap*: **sampling during training and inference should be the exact same**. The analogy is training for a marathon. You prepare by running long distances near the length of the marathon, as opposed to practicing short distances, because you want training to resemble the conditions you will face on race day. By the same logic, a model that never samples during training is preparing for the wrong test. Diffusion and autoregressive models sit especially far from this ideal: both are trained to predict only a single step, yet at inference they are unrolled for hundreds or thousands of steps, so the test-time process is one they never rehearse during training. Under this stricter definition, one-step methods whose training never simulates the sampling process, such as Mean-Flow [9], are not yet end-to-end in the sense we argue for, since they bake in the very train/inference mismatch we want to remove (although, they bake in this mismatch differently than diffusion/autoregressive models, which unroll on their own predictions).

Note that under this definition, contrastively trained models have long been end-to-end: a GAN’s generator samples the same way during training as at inference, and CD-EBMs likewise sample from the model during training. For contrastive models the open problem is covering the full distribution (Section 3.2), not closing a train/inference gap. The hard problem, then, is building *reconstructive* models that are end-to-end, which is difficult for exactly the reason Mode Forcing identifies: training a reconstructive model on its own unrolled sampling collapses the loss minimizer back to the blurred conditional mean (Section 5.4).

Why does this matter? Closing the train/inference gap is ultimately about *generalization*, which we take to be the core goal of the field and turn to next (Section 7.2). It is also worth noting that the sharp split between a “training” phase and an “inference” phase is a feature of how we build models, not an obvious feature of how learning must work; biological learning does not seem to draw the line so cleanly. Concurrent work using Mode Forcing to develop end-to-end Explorative Models [1] demonstrates promising progress toward reconstructive generative models that are end-to-end.

7.2 Generalization, Not Inference Speed

A good deal of recent work optimizes for fewer sampling steps, with consistency models the clearest example [8]. We argue this is the wrong thing to optimize for. Compute grows exponentially over time [69, 70], so inference cost is a shrinking constraint, while what stays scarce is *generalization*: performing well on inputs never seen during training. Models are increasingly limited by generalization rather than by compute. For the two settings where generative models matter most, generalization has a concrete meaning: a world model should predict dynamics correctly in a new environment, and a language model should handle prompts unlike anything in its training set. We expect the next era of generative modeling to be organized around generalization rather than speed. This points back toward end-to-end training, which improves generalization by removing the train/inference gap so the model is evaluated under the conditions it was trained for, and toward energy-based models, which have shown stronger generalization than feed-forward predictors [11, 71].

7.3 Likelihoods and Uncertainty That Actually Work

Another open problem is trustworthy uncertainty. Reliable likelihoods would let a generative model report how confident it is and flag out-of-distribution inputs, which is essential for any system that acts on its own predictions. In practice, today’s likelihoods fall short: they are often poorly behaved out of distribution, sometimes assigning higher likelihood to out-of-distribution data than to the training data itself [72], a failure that persists in modern diffusion models [73, 74]; they are hard to obtain at all for continuous data; and they are only loosely calibrated even for language. We think energy-based models are a promising direction, since they can express relative likelihoods without ever normalizing [11].

7.4 The Infinite-Compute Limit

Finally, a question Mode Forcing raises but does not settle. We have argued that contrastive models have not scaled, plausibly because separating real from contrast in high dimensions seems to require a number of negatives that grows quickly with dimension (Section 3.2). But cost is not the same as impossibility. If compute were effectively unlimited, would contrastive training, which never blurs modes and never suffers exposure bias, become attractive again? We leave this open, but find it a

useful way to stress-test the theory: Mode Forcing explains which models won under the compute budgets we have had so far, and that budget is not fixed.

8 Counterarguments

Mode Forcing makes strong claims, so it invites strong objections. We take the most important ones head-on.

8.1 Scalable Contrastive Models?

The sharpest objection is that contrastive models may be starting to scale after all. GANs remain competitive in places, and recent methods such as Drifting Models [34] revive contrastive training with InfoNCE-style objectives. Some recast a contrastive loss into a more reconstruction-like one using stop-gradients and subtraction, which appears to help training stability, a genuinely promising development.

We are not yet convinced this overturns the picture, for two reasons. First, the sample-complexity issue is a property of the objective, not the implementation: telling real from contrast in high dimensions still appears to need a number of negatives that grows quickly with dimension (Section B.1), and stabilizing the optimization does not remove that. Second, several such methods rely on enumerating or normalizing over discrete conditioning (e.g. class labels), which is unrealistic for the high-dimensional, weakly-conditioned settings that matter most, like video. Whether the stop-gradient trick scales to those settings is, to us, the most interesting open question for contrastive generation, and we would be glad to be proven wrong. Separately, contrastive learning clearly works well for *representations*; our claim is only that it has not served as the primary generative pretraining signal, even if it is useful afterward for steering (classifier-free guidance, reward models).

8.2 Reconstructive Models that Have Not Scaled— Variational Autoencoders (VAEs), Normalizing Flows, and Mixture-Density Networks?

VAEs are reconstructive, so why have they not scaled like diffusion and autoregression? The usual culprit is the gap between the prior and the aggregate posterior, the “prior hole” problem. A VAE samples by drawing a latent from a fixed prior and decoding it, but training only ever decodes latents the encoder produced. As capacity and data complexity grow, the aggregate posterior drifts further from the prior, so at sampling time the decoder is handed latents it never saw during training, and quality degrades. The obvious response, raising the KL weight until the posterior stays matched, trades this failure for the central one of this paper: the KL term taxes the information the latent carries, and that information is all a one-shot decoder has for deciding which mode to commit to. Weaken the penalty and the latent is informative but sampling falls into the holes above; strengthen it and the latent says too little, so the per-latent target stays multimodal and the reconstruction-trained decoder blurs it, the classic blurry VAE sample. As data grows more complex the latent needs to carry more, so the two ends pull apart with scale and no setting of the weight rescues both. This is why VAEs survive today not as standalone generators but as autoencoders that compress images into the latents diffusion models operate on; the widely used SDXL autoencoder, for example, is just the Stable Diffusion latent autoencoder [75, 76]. It is nominally variational, but the KL penalty during training is so slight that its posterior variance nearly vanishes, leaving an effectively deterministic compressor rather than a usable standalone generator [77]. We make this mismatch, and the trade-off above, quantitative in Appendix B.

It is telling that diffusion, viewed as a latent-variable model, is essentially a VAE with both failure points removed [5, 6]: the encoder is the fixed noising process rather than a learned network, so the latent it produces matches the prior by construction and there is no hole to fall into. The decoder is then factorized into many near-unimodal denoising steps conditioned on corrupted real data, so no single prediction carries the multimodality a VAE asks its one-shot decoder to bear.

Normalizing flows [78] and mixture density networks [79] belong to the same story: both are reconstructive and represent a full output distribution directly, so neither blurs in principle, yet neither has scaled to the frontier as standalone approaches. A normalizing flow pays for its exact likelihood with an invertible architecture, and that constraint is mode-shaped: a continuous bijection cannot split its unimodal base into disconnected pieces, so well-separated modes can only be approximated

by leaving thin bridges of density between them, with the map growing arbitrarily ill-conditioned as those bridges thin [80]. The flows that generate well today escape this by importing the tools of the families that did scale, stacking autoregressive transformer blocks, adding noise to the data, and moving into compressed latent spaces [81, 82]; the invertibility remains, but it is the factorization that does the generative work. A mixture density network fails for the most legible reason of the three: its component count is its generative expressivity (Section 4.2), a mode budget fixed at design time and spent on the whole output in a single shot, whereas the equally bounded softmax of an autoregressive model compounds across the sequence through factorization. Once the data carries more modes than components, each component must cover several modes, and its mean lands between them, returning exactly the blur the explicit distribution was meant to prevent; accordingly, where mixtures survive at scale is as per-step heads inside factorized models, expressing the small distribution that remains after conditioning [54]. There may well be other reasons each family lagged, but the pattern across all three is uniform: the VAE survives as a deterministic compressor, the flow becomes competitive only once it factorizes autoregressively, and the mixture lives on as a per-step head. Each family’s surviving form is its mode-forced form, which is why we read their failures to scale as standalone approaches as failures to mode force, rather than as unrelated quirks of architecture.

8.3 Non Contrastive or Reconstructive Models

Moment-matching methods such as MMD-based GMMNs [83] sit at the boundary of our taxonomy: the loss is a discrepancy between p_{data} and p_{θ} rather than a per-sample reconstruction, but it involves no explicit contrast distribution. We treat such methods as reconstructive in spirit, but note that the same underlying machinery becomes contrastive once the kernel is normalized via an InfoNCE-style softmax over positive and negative samples, as in Drifting Models [34]. We do not include moment matching in the main reconstructive family list, as plain MMD-based methods have not scaled in practice.

8.4 Memorization

If reconstructive models are trained to place mass exactly on data points, why do they not simply memorize the training set? Two forces prevent it. The datasets are large enough that memorization is impractical and offers no shortcut to low loss, and neural networks carry inductive biases that favor smooth, generalizing solutions over lookup tables. As a result, “capturing a mode” in practice means landing on the data manifold near real examples, reproducing the structure of the data rather than its exact samples, which is precisely what we want from a generative model.

8.5 What About Other Hypotheses and Arguments?

It’s worth mentioning that Mode Forcing is compatible with other hypotheses, such as the Manifold Hypothesis [28, 29, 84]. Mode Forcing is a predictive theory about modern scalable generative modeling training objectives, whereas the Manifold Hypothesis is a hypothesis regarding the structure of data. While modern generative models are consistent with the Manifold Hypothesis, the Manifold Hypothesis is not *predictive* of which generative models succeed. Recent research demonstrates that Mode Forcing and the Manifold Hypothesis are compatible, where generative models succeed by predicting the data manifold while being mode forced via diffusion during training to prevent mode collapse [84]. The Manifold Hypothesis also explains why the mean of modes is often *not on the data manifold*, as the data manifold is curved, and the Euclidean average of several on-manifold modes generally lands off of it, in a low-density region between the modes rather than on any one of them.

9 Limitations and Conclusion

In this work, we introduced Mode Forcing, the idea that scalable generative modeling comes down to designing a reconstructive objective whose loss minimizer captures modes instead of averaging them. We argued that this single principle explains why each family of generative models won its era, unifies today’s models under two routes (factoring generation or training), and makes predictions that simple experiments and concurrent work bear out.

Limitations. As a position paper, this work has several limitations. First, our mode-count summaries treat blurring as if it were uniform across conditionings and timesteps, when in practice a few hard steps dominate (Appendix B). Second, some supporting arguments, such as why contrastive training has not scaled, are heuristics rather than theorems. Third, the direct experimental checks in this paper are small, controlled ones; larger-scale evidence comes from concurrent work [1, 2]. Finally, reading history through a theory risks fitting the story to the outcome, which is why we lean on the falsifiable predictions of Section 5 rather than on the history alone.

Conclusion. Mode Forcing makes generative models simpler to reason about: new approaches can be classified by how they force modes, blurring can be diagnosed by asking which predictions still face many valid targets, and the open problems we consider most important, truly end-to-end generation and generalization over inference speed (Section 7), follow directly from the theory. The field converged on Mode Forcing one model family at a time; we hope that stating the principle directly lets the next family be designed from it.

Author Contributions

Alexi Gladstone led the project from ideation to execution, developing the theory and method, designing and running all experiments, and writing the paper.

Heng Ji and Yilun Du advised the project throughout, providing invaluable mentorship, feedback on the ideas and writing, and compute support.

Acknowledgement

Huge thanks to Flapping Airplanes for supporting Alexi as a fellow while completing this work. Special thanks to Laude Institute for supporting this work, with a big shoutout to Braden Hancock and K. Tighe. This material is based upon work supported by the U.S. National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 21-46756, U.S. DARPA ECOLE Program No. #HR00112390060, DARPA ITM Program No. FA8650-23-C-7316, NSF Molecule Maker Lab Institute, an AI Institute for Molecular Discovery, Synthesis Strategy, and Manufacturing funded by the U.S. National Science Foundation under Awards No. 2019897 and 2505932, the AI Research Institutes program by National Science Foundation and the Institute of Education Sciences, U.S. Department of Education through Award No. 2229873 - AI Institute for Transforming Education for Children with Speech and Language Processing Challenges, and NSF NAIRR award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, the Defense Advanced Research Projects Agency (DARPA), the Institute of Education Sciences, or the U.S. Department of Education. This research used the Delta and DeltaAI advanced computing and data resources, which are supported by the National Science Foundation (award OAC 2320345 and award OAC 2005572) and the State of Illinois. Delta and DeltaAI are joint efforts of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications. Some of the computations in this paper were run on the FASRC cluster supported by the FAS Division of Science Research Computing Group at Harvard University.

References

- [1] Alexi Gladstone, Yilun Du, and Heng Ji. Explorative modeling: Unlocking a new pretraining axis and end-to-end generative modeling. Preprint, 2026. 1, 3, 7, 8, 9, 10, 11, 12, 15, 18, 26, 28, 29, 30, 31, 33
- [2] Alexi Gladstone, Yilun Du, and Heng Ji. Jumpy generative models: Generation is way easier than you thought. Preprint, 2026. 1, 3, 8, 11, 12, 18
- [3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003. 2, 4, 5, 24
- [4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2, 3, 5
- [5] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015. 2, 4, 5, 16, 24, 25
- [6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 4, 7, 8, 9, 13, 16, 24, 25
- [7] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 2, 5, 8, 9, 14, 25
- [8] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023. 2, 5, 7, 11, 15, 24, 26
- [9] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025. 2, 5, 11, 15, 24, 26
- [10] Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pages 8489–8510. PMLR, 2023. 2, 24
- [11] Alexi Gladstone, Ganesh Nanduru, Md Mofijul Islam, Peixuan Han, Hyeonjeong Ha, Aman Chadha, Yilun Du, Heng Ji, Jundong Li, and Tariq Iqbal. Energy-based transformers are scalable learners and thinkers. *arXiv preprint arXiv:2507.02092*, 2025. 2, 5, 15, 24, 25
- [12] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010. 3
- [13] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 2006.
- [14] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 3
- [15] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 3
- [16] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008. 3
- [17] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on machine learning*, pages 833–840, 2011. 3
- [18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 3
- [19] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 3, 5
- [20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. 3, 5, 14

- [21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 3, 5
- [22] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 3, 5
- [23] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021. 3, 5
- [24] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022. 3, 5
- [25] 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. 3
- [26] 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. 3
- [27] Ruiqi Gao, Emiel Hooeboom, Jonathan Heek, Valentin De Bortoli, Kevin P. Murphy, and Tim Salimans. Diffusion meets flow matching: Two sides of the same coin. 2024. URL <https://diffusionflow.github.io/>. 4, 14
- [28] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009. 4, 17
- [29] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009. 4, 17
- [30] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 5
- [31] Miguel A Carreira-Perpinan and Geoffrey Hinton. On contrastive divergence learning. In *International workshop on artificial intelligence and statistics*, pages 33–40. PMLR, 2005. 5
- [32] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in neural information processing systems*, 32, 2019. 5
- [33] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 5
- [34] Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026. 5, 16, 17
- [35] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 5
- [36] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 5
- [37] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 5
- [38] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. *Advances in neural information processing systems*, 31, 2018. 5
- [39] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62, 2022. 5, 7, 25
- [40] Maciej Wiatrak, Stefano V Albrecht, and Andrew Nystrom. Stabilizing generative adversarial networks: A survey. *arXiv preprint arXiv:1910.00927*, 2019. 5

- [41] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Leo Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 1(8):1, 2024. 5
- [42] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Hugué, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023. 6
- [43] Kilian Fatras, Younes Zine, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Learning with minibatch Wasserstein: asymptotic and gradient properties. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pages 2131–2141, 2020. 6
- [44] Kilian Fatras, Younes Zine, Szymon Majewski, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Minibatch optimal transport distances; analysis and applications. *arXiv preprint arXiv:2101.01792*, 2021. 6
- [45] Samuel Boîté, Julie Delon, and Kimia Nadjahi. Expected batch optimal transport plans and consequences for flow matching. *arXiv preprint arXiv:2605.12174*, 2026. 6
- [46] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024. 6
- [47] Ho Kei Cheng and Alexander Schwing. The curse of conditions: Analyzing and improving optimal transport for conditional flow-based generation. *arXiv preprint arXiv:2503.10636*, 2025. 6
- [48] 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. 6
- [49] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2107.03006. 10
- [50] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2406.07524. 10
- [51] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations*, 2016. 11
- [52] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 11
- [53] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *arXiv preprint arXiv:2406.11838*, 2024. 11
- [54] Michael Tschannen, Cian Eastwood, and Fabian Mentzer. Givt: Generative infinite-vocabulary transformers. In *European Conference on Computer Vision*, pages 292–309. Springer, 2024. 11, 17
- [55] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pages 34668–34684, 2025. 11, 14
- [56] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 12
- [57] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *arXiv preprint arXiv:2201.06503*, 2022. 12, 14
- [58] 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 (NeurIPS)*, 2022. 13
- [59] Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025. 13

- [60] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. 13, 14
- [61] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 13
- [62] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 13
- [63] 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. 14
- [64] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011. 14, 24
- [65] 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. 14
- [66] Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. *arXiv preprint arXiv:2308.15321*, 2023. 14
- [67] Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. *arXiv preprint arXiv:2305.15583*, 2023. 14
- [68] Kaiming He. Towards End-to-End Generative Modeling. Tutorial/Workshop talk, CVPR 2025, 2025. https://people.csail.mit.edu/kaiming/cvpr25talk/cvpr2025_meanflow_kaiming.pdf. 14
- [69] Jaime Sevilla, Lennart Heim, Anson Ho, Tamay Besiroglu, Marius Hobbhahn, and Pablo Villalobos. Compute trends across three eras of machine learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2022. doi: 10.1109/IJCNN55064.2022.9891914. 14, 15
- [70] Richard Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38, 2019. 14, 15
- [71] Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Learning iterative reasoning through energy minimization. In *International Conference on Machine Learning*, pages 5570–5582. PMLR, 2022. 15
- [72] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don’t know? *arXiv preprint arXiv:1810.09136*, 2018. 15
- [73] Hamidreza Kamkari, Brendan Leigh Ross, Jesse C. Cresswell, Anthony L. Caterini, Rahul Krishnan, and Gabriel Loaiza-Ganem. A geometric explanation of the likelihood OOD detection paradox. In *International Conference on Machine Learning*, pages 22908–22935. PMLR, 2024. 15
- [74] Alvin Heng, Alexandre H. Thiery, and Harold Soh. Out-of-distribution detection with a single unconditional diffusion model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 15
- [75] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023. 16
- [76] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 16
- [77] Qifan Li, Xingyu Zhou, Jinhua Zhang, Weiye You, and Shuhang Gu. Taming sampling perturbations with variance expansion loss for latent diffusion models, 2026. Accepted to CVPR 2026. 16
- [78] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *International Conference on Learning Representations*, 2017. 16
- [79] Christopher M Bishop. Mixture density networks. 1994. 16
- [80] Rob Cornish, Anthony L Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows. In *International Conference on Machine Learning*, pages 2133–2143. PMLR, 2020. 17

- [81] Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Josh Susskind. Normalizing flows are capable generative models. In *International Conference on Machine Learning*, 2025. 17
- [82] Jiatao Gu, Ying Shen, Tianrong Chen, Laurent Dinh, Yuyang Wang, Miguel Angel Bautista, David Berthelot, Josh Susskind, and Shuangfei Zhai. Starflow-v: End-to-end video generative modeling with normalizing flows. *arXiv preprint arXiv:2511.20462*, 2025. 17
- [83] Yujia Li, Kevin Swersky, and Richard S. Zemel. Generative moment matching networks. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, JMLR Workshop and Conference Proceedings*, pages 1718–1727. JMLR.org, 2015. URL <http://proceedings.mlr.press/v37/li15.html>. 17
- [84] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 36115–36125, 2026. 17
- [85] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Occam’s razor. *Information Processing Letters*, 24(6):377–380, 1987. 24
- [86] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. doi: 10.1016/0005-1098(78)90005-5. 24
- [87] Peter D. Grünwald. *The Minimum Description Length Principle*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2007. 24
- [88] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. 24
- [89] Aapo Hyvärinen. Some extensions of score matching. *Computational Statistics & Data Analysis*, 51(5): 2499–2512, 2007. 24
- [90] Diederik P Kingma and Yann LeCun. Regularized estimation of image statistics by score matching. *Advances in Neural Information Processing Systems*, 23, 2010. 24
- [91] James Martens, Ilya Sutskever, and Kevin Swersky. Estimating the hessian by back-propagating curvature. In *International Conference on Machine Learning*, 2012. 24
- [92] Saeed Saremi, Arash Mehrjou, Bernhard Schölkopf, and Aapo Hyvärinen. Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*, 2018. 24
- [93] Yang Song, Sahaj Garg, Jiabin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR, 2020. 24
- [94] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 25
- [95] David McAllester and Karl Stratos. Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 875–884. PMLR, 2020. 26
- [96] Carlos Améndola, Alexander Engström, and Christian Haase. Maximum number of modes of gaussian mixtures. *Information and Inference: A Journal of the IMA*, 9(3), 2020. 28
- [97] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015. 32, 33
- [98] Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, 2016. 34
- [99] Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. Distribution matching in variational inference. *arXiv preprint arXiv:1802.06847*, 2018. 34
- [100] Bin Dai and David Wipf. Diagnosing and enhancing vae models. In *International Conference on Learning Representations*, 2019. 34

A Why Create Mode Forcing?

Why introduce another way of thinking about generative models? Our honest answer is that we wanted an account that is *compressive* and *predictive*, one that explains a lot very concisely and is helpful for figuring out what to try next. We first say what we ask of a theory, then revisit existing accounts through that lens.

A.1 What Makes a Theory Good

We judge a theory by two core properties. The first is **compression**: a good theory captures many observations in a short description. This is the central belief of Occam’s razor [85], the long-standing principle that simpler explanations are better. This belief is not random; there is a deep link between compression and **generalization**, our second property: the shortest description consistent with the data tends to be the one that extends correctly to new cases, an idea made precise by Kolmogorov complexity and the minimum description length principle [86, 87]. A short description reduces incidental details and keeps structures that recur, enabling generalization to new ideas.

Gravity is a great example of an intuitive and compressive theory that generalizes well. Newton’s law writes an enormous range of observations, falling apples, ocean tides, and the orbits of moons and planets, as a single short expression, and precisely because it is so compressed, it generalized far beyond its origins, even predicting unseen planets before they were observed. A good theory of generative modeling should aim for the same: explain the models we have with a short description, and in doing so predict the ones we do not yet have.

Compression also pays a practical dividend: a theory simple and concise enough to hold in one’s head is one researchers can actually apply, extend, and use to design new methods, which we argue is much of the point of a predictive theory. We designed the primary thesis of Mode Forcing with these properties in mind: *design a reconstructive training objective where the loss minimizer captures modes*. Because this single sentence spans autoregression [3], diffusion [6], single-step generative models (Consistency [8] and Mean-Flow [9]), and reconstructive Energy-Based Models [10, 11], and predicts current characteristics of generative modeling as well as new approaches (Section 5), we argue it does what a good theory should: compress what we already know, and predict what we do not (more on the prediction portion is in Section 5.1).

A.2 On Existing Theories

We felt that existing theories of generative modeling, for all their rigor, were better at telling us whether a method is sound than at predicting which methods would scale. It is certainly not that the field lacks mathematical treatment: VAEs have the ELBO, GANs have a minimax analysis, and diffusion has variational bounds along with deep connections to score matching. These treatments are rigorous, and we lean on several ourselves; some even predate their methods, as when diffusion grew out of nonequilibrium thermodynamics [5]. Our concern is only with what they select for. Each shows that a method is *valid*, that in some limit it estimates the right thing, but validity is shared by the approaches that scaled and by those that did not. VAEs optimize the same kind of variational bound as diffusion models and did not scale, and exact likelihood, the most formally grounded objective of all, does not separate them either: autoregressive models and normalizing flows both maximize one, yet only autoregression scaled.

Score-based models are the clearest example. Learning a distribution through its score, the gradient of its log density, was formalized in 2005 [88], and over the following decade and a half the field built a series of increasingly practical estimators for it. The original objective required second derivatives of the log-density, which scale poorly, so gentler variants followed: ratio matching for discrete data [89], regularized score matching for natural images [90], curvature propagation to cheapen the Hessian terms [91], denoising score matching at a single noise level [64, 92], and sliced score matching, which finally scaled to deep networks and high dimensions [93]. Every one of these was sound, and several proved useful for tasks like denoising and super-resolution, yet none produced a strong generative model: the single-noise energy estimators produced no samples at all [92], and the sliced variant’s generative demonstrations stayed at the scale of MNIST and small face images [93]. What changed was not a better estimator but a reshaped target: corrupt the data with noise and train the network to undo it, turning score estimation into a reconstruction problem [64], and do so across many noise

scales rather than one [5, 94], the recipe diffusion models made ubiquitous [6] (the 2015 form had the right shape but weak samples until noise schedules and sampling matured). The researchers behind the sliced estimator gave the diagnosis themselves: raw scores are ill-defined off the data manifold and poorly estimated in low-density regions, and Langevin sampling mixes badly between separated modes, while perturbing the data at many noise levels, with sampling annealed across them, addresses all three at once [94]. It is also exactly the factorization into near-unimodal pieces that Mode Forcing points to. We do not think the score formalism was wrong, and it is possible that a sharper reading of it could have anticipated which variants would generate well. Our point is gentler: the formalism treats a model trained at one noise level and one trained across many as the same kind of objective, so on its own it offered little reason to prefer the variant that worked, whereas the mode-forcing lens makes that preference immediate.

Mode Forcing draws a finer line. Compare flow matching with a naive one-shot regressor that maps noise straight to data and blurs (Section 4.2). To the standard toolkit the two are the same kind of least-squares problem: both regress onto a conditional target, both minimizers are conditional means, and the guarantees concern the estimation rather than the shape of the conditional being averaged. Flow matching does carry a guarantee of its own, that regressing per-sample velocities recovers the correct marginal field [7], but it tells us which mean each step computes, not whether that mean lands between modes; the near-unimodality of the per-step target is a separate, geometric fact. The one-shot regressor is even a flow model in miniature, trained at a single time step and sampled in one jump (Section 5.1). What separates the two is only how finely the target is factorized, the property Mode Forcing tracks and the standard analyses do not.

None of this is an argument against formalization. Maximum likelihood, the ELBO, and their relatives answer a real question, whether an estimator is sound, but *that has not been the question that decides which methods succeed*. We argue that math explaining or guiding progress is most useful when it is *concise and predictive*, able to separate the designs that will scale from the equally valid ones that will not. The field could use more mathematical tools whose representations make a method’s working principle immediate, rather than leaving it implicit in a derivation. Mode Forcing is our attempt at such an account, stated intuitively and formally (Section 4.2, Appendix B).

B Mode Forcing: Formal Treatment

This appendix is the formal counterpart of Section 4.2. We first define the two families of generative models precisely, then make the mean-blurring claim quantitative: reconstructive losses produce mode-blurred predictors, and each of the two routes of Section 4.2 removes the blur by keeping any single prediction from being forced to average across modes. We then work out what the explorative route optimizes, and end with the failure the framework attributes to VAEs, the reconstructive family that did not scale.

B.1 The Two Families

Definition B.1 (Contrastive Generative Model). *A contrastive generative model is a generative model whose training objective $\mathcal{L}(\theta)$ depends jointly on samples from the data distribution $p_{\text{data}}(\mathbf{x})$ and samples from a contrast distribution $q(\mathbf{x})$ (e.g., a generator’s output, a noise distribution, or model-generated samples), and whose loss minimizer is determined by some discrepancy between p_{data} and q rather than by reconstruction of any specific portion or form of a training sample. A common (but not exhaustive) form is*

$$\mathcal{L}(\theta) = \mathbb{E}_{p_{\text{data}}}[f(\mathbf{x}; \theta)] - \mathbb{E}_q[g(\mathbf{x}; \theta)], \quad (1)$$

under which optimality typically requires the model to capture a monotonic transformation of the density ratio p_{data}/q (as for GAN discriminators and InfoNCE-style critics), or, when q is the model’s own distribution (as in contrastive divergence), to drive that distribution toward p_{data} .

Why contrastive training has not scaled (heuristic). Definition B.1 also points to why contrastive models have struggled at scale. The objective only constrains the model where contrast samples land: a region the negatives never visit is a region where mass can be misplaced for free. The volume of such regions grows exponentially with dimension, so the number of negatives needed to witness every place the model could go wrong grows at a similar rate [11, 39]. A rigorous fragment of this intuition

exists for InfoNCE-style objectives: any distribution-free, high-confidence lower bound on mutual information built from K samples cannot exceed $O(\log K)$ [95], so certifying an amount of structure that grows with dimension requires a number of negatives exponential in dimension. We present this as a heuristic rather than a theorem for generative modeling: in particular, it is open whether the relevant dimension is the ambient one or the much smaller intrinsic dimension of the data manifold. Either way the requirement grows exponentially in the relevant dimension, which is plausibly large for natural data, matching the empirical difficulty of scaling contrastive generation; a precise rate is left to future work.

Definition B.2 (Reconstructive Generative Model). *A reconstructive generative model is a generative model whose training objective minimizes a divergence between the model’s outputs and targets drawn from $p_{\text{data}}(\mathbf{x})$, where the loss minimizer is determined entirely by p_{data} rather than by a separate contrast distribution. A canonical instance, given a coupling $p(\mathbf{c}, \mathbf{y})$ over (conditioning, target) pairs, a reconstruction loss J , and a model $f_\theta : \mathcal{C} \rightarrow \mathcal{Y}$, takes the form*

$$\mathcal{L}(\theta) = \mathbb{E}_{(\mathbf{c}, \mathbf{y}) \sim p(\mathbf{c}, \mathbf{y})} [J(\mathbf{y}, f_\theta(\mathbf{c}))], \quad (2)$$

where the target $\mathbf{y}$ may be a full training sample, a portion of one (as in masked or autoregressive prediction), or a transformation thereof (as in diffusion), possibly involving the model’s own predictions (as in the bootstrapped targets of Consistency and Mean-Flow models [8, 9]). When J is a Bregman divergence (e.g., squared error), the minimizer at $\mathbf{c}$ is $\mathbb{E}[\mathbf{y} \mid \mathbf{c}]$, the fact the rest of this appendix builds on. This covers cross-entropy as well: against one-hot targets it is the Bregman divergence of negative entropy, and the mean of one-hot targets is the full conditional distribution, which is why a softmax output preserves multimodality rather than blurring it (Section 4.2).

B.2 Quantifying Mode Blurring

Mode count. For a distribution q on $\mathcal{Y} \subseteq \mathbb{R}^d$ with density $q(y)$, define the *mode count*

$$M(q) \triangleq |\{y \in \mathcal{Y} : \nabla q(y) = 0, \nabla^2 q(y) \prec 0\}|,$$

the number of strict local maxima of q (for an atomic distribution, the cardinality of its support). Three conventions hold throughout: the relevant conditional densities are assumed to exist, reconstruction losses are Bregman divergences, and M counts modes irrespective of their mass or separation, so a faint bump and a dominant peak each count once; these are the regularity assumptions the analysis runs on, and the ones the companion paper inherits [1].

Modes at the loss scale. Counted this literally, M is brittle: it registers arbitrarily small wiggles, and data supported on a lower-dimensional manifold has no ambient density to count at all. When this matters we read counts *at the loss scale*: $M_\sigma(q) \triangleq M(q * k_\sigma)$, the mode count after smoothing by the Gaussian kernel k_σ that the reconstruction loss itself corresponds to (squared error is, up to constants, the negative log of a Gaussian of width σ). M_σ is finite and stable for natural data, and the mode-separation condition of Proposition B.5 starts from resolvability at this scale, strengthened there by a margin depending on K and the smallest mode’s mass. The continuous-target statements of this appendix survive with M read as M_σ whenever the modes in question are resolvable at scale σ ; discrete distributions keep the atomic convention, since cross-entropy carries no such scale. Where a target’s own mode count matters we write $M^*(c) \triangleq M_\sigma(p^*(\cdot \mid c))$, the number of distinct valid answers the data has at c , and drop the argument when the conditioning is fixed.

Mean-blurring lemma. Consider any reconstructive loss $\mathcal{L}_{\text{rec}}(f) = \mathbb{E}_{\mathbf{c}, \mathbf{y} \sim p^*(\cdot \mid \mathbf{c})} [J(\mathbf{y}, f(\mathbf{c}))]$, where p^* writes the data distribution (p_{data} above) in conditional form and J is a Bregman divergence taken with the target in its first argument (e.g. squared Euclidean distance, $J(a, b) = \|a - b\|^2$). The unique minimizer is the conditional mean, $f^*(c) = \mathbb{E}[\mathbf{y} \mid c]$. The induced model distribution $P_{f^*}(\cdot \mid c) = \delta_{f^*(c)}$ is a point mass, so $M(P_{f^*}(\cdot \mid c)) = 1$ for all $c \in \mathcal{C}$, regardless of how many modes the true conditional $p^*(\cdot \mid c)$ has. Rich conditioning narrows $p^*(\cdot \mid c)$ but does not generically make it unimodal (e.g. $c = \text{"dog"}$ remains highly multimodal), so the collapse is not escaped by conditioning alone. One corollary is worth noting: at the very first step of a denoising trajectory the input is pure noise and carries no information beyond c , so the optimal prediction at that step is determined by the conditional mean $\mathbb{E}[\mathbf{y} \mid c]$ alone (exactly $\mathbb{E}[\mathbf{y} \mid c]$ when the model predicts data, minus the noise input when it predicts velocity), the prediction Section 5.3 verifies empirically.

Remark: the ceiling is set by determinism, not by the mean. The Bregman assumption determines *where* the single prediction lands, not that there is only one. Any deterministic predictor induces the point mass $P_f(\cdot | c) = \delta_{f(c)}$, so direct regression has $E = 1$ under *every* reconstruction loss: L_1 yields the coordinatewise conditional median, quantile losses yield conditional quantiles (again coordinatewise), and a perceptual or otherwise robust loss still commits to some single point. The Bregman mean generically lands off the data manifold between separated modes, whereas a robust loss can land its point on one of them, trading blurring for mode dropping; either way every other mode is lost. Swapping the loss therefore cannot raise the ceiling; only reshaping the objective can.

Generative expressivity. The lemma proves something stronger than “the mean blurs”: it caps how many modes a direct regressor can ever produce. Define the *generative expressivity* of an approach as the largest conditional mode count its loss minimizers can ever retain, over any data distribution,

$$E \triangleq \sup_{p^*, c} \sup_{\theta^* \in \arg \min_{\theta} \mathcal{L}(\theta)} M(P_{\theta^*}(\cdot | c)),$$

where $P_{\theta}(\cdot | c)$ is the distribution induced by the model’s inference-time sampling procedure, the supremum over data distributions p^* makes E a property of the training objective itself rather than of any particular dataset, and minimizers range over all densities (we assume the nonparametric optimum is realizable). This is the third ceiling of Section 4.2, alongside parameter expressivity (the functions a network can represent) and data expressivity (the structure a dataset shows the model). Where those two bound how *well* a model fits a target, generative expressivity bounds how many modes its output can carry *at all*. By the mean-blurring lemma, end-to-end direct regression under a Bregman loss has $E = 1$: its minimizer is unique for every p^* , a point mass for every c , and because that minimizer is the *unconstrained* optimum, enlarging the network only approaches the same one-mode optimum and never passes it. Generative expressivity is therefore a bottleneck distinct from the other two, in that no amount of parameters or data lets a direct regressor capture a second mode, much as no amount of training lets a linear model fit a parabola.

The two routes of Section 4.2 each raise this ceiling. Factoring composes ceilings across pieces when each piece commits to one of finitely many values: a softmax over a vocabulary of size V leaves V choices per step, so an autoregressive chain of length L reaches V^L . A head with continuous support does not compose this way, since it passes on a continuum that a later step can fold onto many separated modes, so for those we read the count off the whole sampler rather than multiplying per-piece ceilings. A standalone C -component mixture, for its part, retains C modes of its own target, counted at the loss scale. Denoising raises E within the same route but differently, since each step outputs a bare mean: conditioning every step on the running noisy state makes the composed noise-to-data map genuinely depend on the noise, so its pushforward can be multimodal and E becomes very large, read as always at the loss scale (Section B.3); what the number of steps controls is how faithfully mass is placed on those modes, not the count itself. Exploring raises E without factoring: with K stochastic candidates, every minimizer on a target of up to K well-separated modes places mass on every one of them, so $E \geq K$, with the growth set by the objective rather than the output parametrization. Proposition B.5 proves this for the smooth form of the objective: the separation must be wide enough that the penalty for leaving a mode uncovered, which grows with the squared separation and that mode’s mass, dominates the at-most-log K gain of concentrating elsewhere. In every case the count is set by how the objective is factored or how each piece is parametrized.

The unfactored families of Section 8 read off directly. A standalone mixture density network is the $T = 1$ explicit-distribution case, so its ceiling $E = C$ is fixed at design time, while normalizing flows and other exact-likelihood models carry a ceiling as large as flow’s, their binding constraint being architectural rather than a ceiling of this kind. We read E off reconstructive objectives only. A contrastive optimum is fixed by a discrepancy between the model and a contrast distribution rather than by any per-prediction target, so this accounting does not apply to it; the difficulty those objectives face instead, that mass goes unconstrained wherever the negatives never visit, is the separate heuristic under Definition B.1.

What E separates, and what it does not. E is an at-optimum quantity, and for a *consistent* factored objective the optimum carries no deficit: autoregression under cross-entropy and continuous-time flow sampled exactly admit minimizers whose sampling distribution is p^* itself, so at their optima E

reaches the most the sample space allows. E therefore distinguishes direct regression from factored models, but not one factored family from another [1]. What does separate them lives one level down, at each single prediction: every piece of a factored objective carries a small ceiling of its own (the per-piece e above; one mode for a squared-error step) and a residue of its own, the per-prediction residual multimodality M_t of Section B.3, counting how many valid targets still compete for that one prediction—a residue that richer conditioning and finer factoring shrink without ever fully removing. The companion paper [1] names these *per-prediction expressivity* and *residual multimodality*, and builds its method on attacking the residue directly rather than factoring it smaller.

Expected mode count and blurring. Define the expected (over conditioning) conditional mode count $\bar{M}(q) \triangleq \mathbb{E}_c[M(q(\cdot | c))]$. By the lemma, $\bar{M}(P_{f^*}) = 1$, whereas a generally multimodal target has $\bar{M}(p^*) \geq 1$, with strict inequality whenever $p^*(\cdot | c)$ is multimodal with positive probability over c . We define the *expected mode blurring*

$$\mathcal{B}(q) \triangleq \bar{M}(p^*) - \bar{M}(q),$$

the average number of true modes lost per conditioning input (nonnegative whenever the model’s realized count does not exceed $\bar{M}(p^*)$, the regime of interest here). Generative expressivity lower-bounds this loss: at optimum a model’s realized mode count cannot exceed its ceiling, $\bar{M}(P_{\theta^*}) \leq E$ for every minimizer θ^* , so an approach with $E < \bar{M}(p^*)$ must leave residual blurring $\mathcal{B} \geq \bar{M}(p^*) - E$. Plain L_2 regression is the extreme case $E = 1$, incurring the maximal blurring $\mathcal{B}(P_{f^*}) = \bar{M}(p^*) - 1$.

B.3 The Per-Prediction Ledger

E is a property of a whole sampler, and the deficit that separates one factored family from another sits one level down, at each single prediction. We make that level precise here: every prediction carries a ceiling of its own, faces a demand of its own, and blurs whatever part of that demand its ceiling cannot cover.

Setup. A factored approach makes T predictions per sample. Write c_t for what prediction t conditions on and $p_t^*(\cdot | c_t)$ for the conditional over its own target: for a denoising model c_t is the running noisy state together with c and the time, for a masking model the elements already revealed together with c , and for a single-shot model $T = 1$ with $c_1 = c$, so one bookkeeping covers all three. Each prediction is fit by its own loss $\mathcal{L}_t(\theta) = \mathbb{E}_{c_t, y_t}[J(y_t, f_t(c_t))]$ and turns the network’s output into a distribution $Q_{\theta, t}(\cdot | c_t)$ over that target through its output parametrization: a point mass for a bare mean, a categorical for a softmax, a mixture for a mixture head. Continuous counts are read at the loss scale as in Section B.2, while discrete predictions keep the atomic convention.

Per-prediction expressivity. The ceiling one prediction carries is read exactly as E is, with the count taken over that prediction’s own target and the minimizer over that prediction’s own loss,

$$e_t \triangleq \sup_{p^*, c_t} \sup_{\theta^* \in \arg \min_{\theta} \mathcal{L}_t(\theta)} M(Q_{\theta^*, t}(\cdot | c_t)),$$

the largest number of its target’s modes prediction t ’s minimizer can retain, over any data distribution and any conditioning it might be asked at. Like E , this reads the ceiling at the optimum rather than over all weights, which is what keeps it a property of the objective rather than of the parametrization alone: a prediction is credited with the modes its own loss drives it to keep, so attaching a latent to a head that is still fit by a bare mean raises nothing, while scoring K draws against the nearest target does. The definition takes each piece to be minimized on its own, the same nonparametric idealization E makes one level up, and a single network shared across every t need not respect it; we read e_t under that idealization throughout. The values are the ones Section B.2 reads off: $e_t = 1$ for a bare-mean step under any Bregman loss, V for a softmax over a vocabulary of size V , and C for a C -component mixture head, this last provided the components are separated at the loss scale, since in two or more dimensions a mixture can carry more maxima than it has components even with a shared isotropic covariance [96].

Residual multimodality. Against that ceiling stands the demand, how many valid targets compete for the prediction when it is asked at a particular c_t ,

$$M_t(c_t) \triangleq M(p_t^*(\cdot | c_t)).$$

We keep this pointwise rather than averaging over c_t , because what a prediction pays is $\max(M_t(c_t) - e_t, 0)$ at each c_t , and the average of that shortfall is at least the shortfall of the average. Averaging first therefore understates the residue, and by enough to matter: a prediction whose target is unimodal most of the time and badly multimodal on the rest can blur heavily while its average count sits near one, which is exactly what a noise schedule or a masking order produces. At $T = 1$ the definition recovers the whole-output count, $\mathbb{E}_c[M_1(c)] = \bar{M}(p^*)$, so the single-shot statements are the one-prediction case of the same ledger.

Noise inputs do not raise the ceiling. One consequence of the mean-blurring lemma is worth spelling out, since the fix it rules out is the one most often reached for first. Take end-to-end regression with a noise input attached, $f_\theta(c, z)$ with z drawn independently of (c, y) and trained under $\mathbb{E}_{c,y,z}[\|y - f_\theta(c, z)\|^2]$. Because z carries no information about the target, the minimizer at each pair is $f^*(c, z) = \mathbb{E}[y \mid c, z] = \mathbb{E}[y \mid c]$, the same conditional mean for every draw, so the induced sampling distribution is the point mass it was without the noise and $E = 1$ still. What separates this from a denoising model is not the presence of noise but whether the loss pays for spending it: flow conditions each target on the very noise the sampler will draw, so at the optimum the map genuinely depends on z , while exploration scores K draws against the nearest target, so distinct draws are rewarded for answering for distinct modes.

Counts are read at a resolution. Mode counts are read throughout at the loss scale σ of Section B.2, for the randomness a sampler draws on as much as for the data, so two outputs the loss cannot tell apart count once here; the companion paper’s report of an unbounded E for continuous diffusion [1] is the $\sigma \rightarrow 0$ reading of the same quantity. This is a convention for making counts comparable rather than a bound on them, and it is worth being plain that finite is not the same as binding. For every family that has scaled, any ceiling one can write down sits astronomically above the mode count of any distribution anyone fits, exactly as V^L does for a language model, so E should not be read as the reason models struggle when the data carries many modes. What E earns is the separation it was introduced for, between direct regression, whose ceiling is genuinely one, and everything that reshapes the objective. What binds past that point is the per-prediction shortfall $M_t(c_t) - e_t$, which is a statement about one prediction and does not shrink as the chain gets longer. Two observations already point toward a capacity that is not merely finite but modest. Conditioning a model more narrowly helps far more than a ceiling with this much headroom would suggest, and adding expressivity to recipes whose sampling has already converged keeps paying, by margins that grow rather than shrink as data and parameters scale [1]. Both are what one expects if the binding quantity is how many modes a trained model actually retains rather than how many its objective permits, and we measure that realized count directly in the full version of this work, sweeping the factorization upward from the one-step case where the lemma above pins it exactly at one.

B.4 How Each Route Removes the Blur

The two routes of Section 4.2 each drive $\mathcal{B}$ toward zero: the first factors the objective, the second explores it.

1. **Factor the objective (denoising and masking).** Factoring splits one prediction into T , and it works because each of the T targets carries less choice than the whole did. Two residues have to be kept apart here, since they answer to different knobs. The *sampler residue* is how much choice remains inside a single transition of the true chain, $S_t(c_t) \triangleq M(p^*(\cdot \mid y_{t-1}, c))$ read pointwise in (c, y_{t-1}) ; it shrinks as the steps get finer, and at the loss scale it reaches one exactly once a step’s spread falls below σ , since a transition the loss cannot resolve leaves nothing to choose between. The *predictor residue* is the M_t of Section B.3, how many valid targets compete for the network’s single output while it is being trained, which for a denoising model is the conditional over whatever that step regresses, the clean endpoint or the velocity, and which is set by the noise level and the data rather than by the number of sampling steps. A bare-mean head does blur across the second of these; the chain survives that blur because a blurred endpoint is still a usable velocity, so the choice it declined to make is carried by the initial draw and the trajectory that draw determines rather than by the prediction itself. Masking models shrink both residues the same way, conditioning each element on those already revealed. Where a piece remains multimodal, an explicit-distribution output preserves it instead of averaging it: the per-piece loss minimizer is then

the *full* conditional distribution rather than its mean, so $\bar{M}(P_\theta) \rightarrow \bar{M}(p^*)$ as the fitted conditionals approach the true ones. In the fine-factorization limit the two parametrizations coincide: a chain of near-unimodal conditionals, expressed either as Gaussians (reduced to mean prediction) or as explicit distributions.

2. Explore the objective (winner-take-all selection; XM). Under

$$\mathcal{L}_{\text{XM}}(\theta) = \mathbb{E}_{c, y \sim p^*(\cdot|c)} \mathbb{E}_{\hat{y}_1, \dots, \hat{y}_K \stackrel{\text{iid}}{\sim} G_\theta(\cdot|c)} \left[\min_{k \in \{1, \dots, K\}} \|\hat{y}_k - y\|^2 \right], \quad (3)$$

only the nearest of K stochastic candidates is penalized, so the min acts like a k -means assignment: each draw is answerable only for the region of $p^*(\cdot|c)$ it lands nearest to, and averaging is confined within that region rather than running across the whole target. What K draws leave uncovered is therefore counted one conditioning at a time,

$$\mathcal{B} \approx \mathbb{E}_c [\max(M^*(c) - K, 0)],$$

with the maximum inside the expectation and not outside it, since a conditioning whose target already fits inside K cannot lend its slack to one that does not. The estimate is two-sided and heuristic: Proposition B.5 pins the case $M^*(c) \leq K$, where every mode is retained and the term vanishes, while beyond K coverage simply stops being enforced and nothing is proven in either direction. As K grows the regions shrink until every mode is answered for, so $\mathcal{B} \rightarrow 0$. With $K = 1$ the assignment is trivial and the loss reduces to the L_2 regression of the lemma. Equivalently, selecting which candidate to reinforce repairs the random coupling of Section 3.4: the model chooses which target each draw answers for, instead of averaging over the pairing it was handed.

Two refinements to the exploration picture. Two consequences of Proposition B.5 sharpen route 2. First, at finite K the optimum is not simply one candidate per mode: because candidates are sampled rather than placed, a draw can miss a mode entirely, so minimizers keep some insurance mass *between* modes to soften the cost of a miss, a bridge that fades once candidates sufficiently outnumber the modes. Second, exploration does not overshoot: on a target with $M^* \leq K$ well-separated modes, minimizers retain all M^* of them, and once the bridge fades the model’s density carries exactly those maxima when read at the loss scale; unimodal targets, where no draw can miss, keep unimodal optima at any K .

Hard min versus smooth form. The hard min displayed above is what is implemented; the smooth form scores the K candidates by the average of their Gaussian kernels rather than by the best alone, and differs from the hard min by at most $\log K$. The clean distributional statements belong to the smooth form—there Forward exploration performs maximum likelihood of its K -candidate mixture at every K (Proposition B.3), while the Reverse variant targets the reverse KL and collapses without an entropy bonus (Proposition B.4)—whereas the hard min’s large- K limit guarantees coverage of the data’s support without controlling how mass is spread over it; Section B.5 states and proves all three.

A floor, and where the shortfall is paid. Whatever ceiling an approach carries bounds what any of its minimizers can retain, which gives a floor on the blurring one conditioning at a time,

$$\mathcal{B} \geq \mathbb{E}_c [\max(M^*(c) - E, 0)],$$

in the regime where the model produces no modes the data does not have. The placement of that maximum is not a technicality. Since $x \mapsto \max(x - e, 0)$ is convex, pulling it outside the expectation does not merely loosen the bound but can zero it out: a target carrying three modes half the time and one the other half has $\bar{M}(p^*) = 2$, so against a ceiling of two the outside form reads nothing lost, while the model in fact retains at most one and a half. The same reading applies one level down. A prediction pays $\max(M_t(c_t) - e_t, 0)$ at each conditioning it is actually asked at, and averaging its target’s mode counts before taking that shortfall understates what it gives up, by enough to matter wherever a schedule mixes easy conditionings with hard ones. Both routes act on that shortfall, richer conditioning shrinking M_t while explicit heads and exploration raise e_t , which is why exploration helps most where the residue is largest [1]. We do not claim the per-prediction shortfalls add to a total: mode counts compose multiplicatively where they compose at all, so a summed budget is an approximation that holds only while every deficit is small.

One qualifier: when a step reveals several elements at once through a product head (as in parallel unmasking), e_t must be read *jointly*. Each marginal is retained exactly, but a product can only place joint modes on the axis-aligned grid of its marginals' modes, so the dependence among simultaneously revealed elements is lost, a failure the count alone understates. This is part of why revealing many positions per step degrades quality, and it is why exploration helps masked models most: the K latent-conditioned candidates turn the step's product head into a mixture, which can carry the dependence a single product cannot [1].

Caveat: what averaging still hides. The summaries $\bar{M}$ and $\mathcal{B}$ aggregate across c and, for iterative methods, across trajectory steps t , and what they hide is the shape of the deficit rather than its size: two schedules losing the same number of modes can place the losses very differently, and placement matters, since a mode lost early in a trajectory is never recovered while one lost late costs a single detail. The counts are also blind to mass, treating a faint bump and a dominant peak alike, so they say nothing about how well mass is spread over the modes a model does keep, which is where a consistent factored objective spends most of its remaining error. A refined analysis would track the joint distribution of $M_t(c_t)$ over (t, c_t) together with the mass on each mode. We use these summaries as first-order accounting with both limits in mind.

B.5 What Exploration Optimizes

Overview. This subsection makes route 2 precise, and involves two distributions: the data p^* and the model's generations G_θ . Because squared error is, up to a constant, the negative log of a Gaussian of width σ , the same kernel the mode counts of Section B.2 are read at, the loss blurs whichever distribution it is applied to, so we write $P_{\theta,\sigma}$ for the model's generations blurred by this Gaussian and p_σ^* for the data blurred the same way. In their smooth forms, Forward and Reverse exploration minimize

$$\text{KL}(p^* \parallel P_{\theta,\sigma}) + H(p^*) \quad \text{and} \quad \text{KL}(G_\theta \parallel p_\sigma^*) + H(G_\theta),$$

respectively, where H is entropy. The two differ only in the entropy each carries. For Forward exploration it is the *data's*, a constant the model cannot change, so Forward exploration performs maximum likelihood, of its K -candidate mixture, at every K . For Reverse exploration it is the *model's* own, which the model can lower by shrinking its spread, so Reverse exploration drifts toward collapse by itself and needs an entropy bonus to become the pure reverse KL.

Setup. We ignore additive constants throughout, as they change neither gradients nor minimizers, and suppress the conditioning c , as every statement holds per condition. The loss between a datapoint y and a generation $\hat{y}$ is $J(y, \hat{y}) = -\log k_\sigma(y, \hat{y})$, where for squared error the kernel is a Gaussian centered on the generation, $k_\sigma(y, \hat{y}) = \mathcal{N}(y; \hat{y}, \sigma^2 I)$, symmetric in its two arguments so that blurring by it is ordinary convolution. Blurring each side gives

$$P_{\theta,\sigma} = G_\theta * k_\sigma, \quad p_\sigma^* = p^* * k_\sigma,$$

so $P_{\theta,\sigma}$ is the model as the loss sees it. This is not the sampling distribution P_θ of Section B.2: an end-to-end explorative model deploys by drawing a single candidate, so $P_\theta = G_\theta$, and counting the modes of $P_{\theta,\sigma}$ is counting the modes of what the model samples read at the loss scale, $M_\sigma(P_\theta)$. Exploration can search in either direction. Forward exploration fixes a datapoint and searches its own generations, which is Equation 3; Reverse exploration fixes one generation and searches the data,

$$\mathcal{L}_{\text{Reverse}}(\theta) = \mathbb{E}_{\hat{y} \sim G_\theta} \mathbb{E}_{y_1, \dots, y_K \stackrel{\text{iid}}{\sim} p^*} \left[\min_{k \in \{1, \dots, K\}} \|y_k - \hat{y}\|^2 \right], \quad (4)$$

erring toward precision where the forward direction errs toward coverage. Finally, the statements below are for the smooth form of exploration, which scores the K candidates by their average kernel, $-\log \frac{1}{K} \sum_k k_\sigma(y, \hat{y}_k)$, rather than by the best alone. This differs from the hard min by at most $\log K$.

Proposition B.3 (Forward exploration is maximum likelihood). *At every K , the smooth Forward objective, the soft counterpart of Equation 3,*

$$L_F(\theta) = \mathbb{E}_{y \sim p^*} \mathbb{E}_{\hat{y}_{1:K} \sim G_\theta} \left[-\log \frac{1}{K} \sum_k k_\sigma(y, \hat{y}_k) \right]$$

is exactly the expected negative log-likelihood of the data under the mixture of the model's own K explored generations, $\frac{1}{K} \sum_k k_\sigma(\cdot, \hat{y}_k)$, so Forward exploration performs maximum likelihood at every K . As $K \rightarrow \infty$ this mixture converges to $P_{\theta,\sigma}$, so minimizing L_F becomes minimizing $\text{KL}(p^ \parallel P_{\theta,\sigma})$, recovering p^* as $\sigma \rightarrow 0$.*

Proof sketch. For a fixed draw of candidates, $\frac{1}{K} \sum_k k_\sigma(\cdot, \hat{y}_k)$ is a normalized density, so the inner term is exactly the negative log-likelihood of y under this mixture, and subtracting the constant data entropy $H(p^*)$ leaves the forward KL to it. The same average is an unbiased estimate of $P_{\theta, \sigma}(y)$, so by Jensen L_F upper-bounds $\mathbb{E}_{p^*}[-\log P_{\theta, \sigma}]$ for every K , tightening as K grows by the argument of IWAE [97] and converging by the law of large numbers under domination, where $\mathbb{E}_{p^*}[-\log P_{\theta, \sigma}] = \text{KL}(p^* \parallel P_{\theta, \sigma}) + H(p^*)$. $\square$

Remark: the optimum at fixed σ . Over all generator densities, the optimum is the KL projection of p^* onto the blurred family $\{G * k_\sigma\}$; it equals p^* exactly when the σ -deconvolution of p^* exists as a density, and in general the match becomes exact only as $\sigma \rightarrow 0$.

The role of K . Since Forward exploration performs maximum likelihood at every K , what K controls is the density each draw fits. At $K = 1$ that density is a single Gaussian of width σ , whose best fit is a point mass at the conditional mean, the mean-blurring lemma read as a likelihood statement. At larger K it is a K -component mixture that can place mass on many modes, becoming $P_{\theta, \sigma}$ as $K \rightarrow \infty$. At inference, though, the model draws a single sample from G_θ , which the loss scores only through its blur, and from that fixed density’s view L_F is an upper bound on its negative log-likelihood that tightens monotonically in K [97]. The gap between the two views is a Jensen gap: because the mixture’s components are sampled rather than placed, finite K penalizes draws that miss a mode. When per-draw scores concentrate this gap is $O(1/K)$ and merely favors slightly lower-variance generators; with well-separated modes and few candidates, misses are costly enough that minimizers hedge mass between modes (Proposition B.5), an effect that fades as K outgrows the mode count. Exploration therefore turns a unimodal regressor into a multimodal likelihood model.

The hard min. The hard min actually implemented differs from the smooth one by at most $\log K$, and the difference matters most in the limit. As $K \rightarrow \infty$ the hard objective asks only that every datapoint have some generation arbitrarily close to it, so any model whose generations cover the data’s support becomes optimal, regardless of how it spreads mass over that support. The hard min is therefore a *coverage* objective, sharing the mode-covering character of the forward KL without pinning down the density, which is why the clean distributional statements are those of the smooth form. This is also why the $E \geq K$ of Section B.2 and the route-2 estimate are smooth-form statements: at finite K the two objectives sit within $\log K$ of each other, the same budget the separation condition of Proposition B.5 is asked to dominate, but in the large- K limit only the smooth form still controls where mass goes.

Proposition B.4 (Reverse exploration targets the reverse KL, but collapses without an entropy term). *At every K , the smooth Reverse objective, the soft counterpart of Equation 4,*

$$L_R(\theta) = \mathbb{E}_{\hat{y} \sim G_\theta} \mathbb{E}_{y_{1:K} \sim p^*} \left[-\log \frac{1}{K} \sum_k k_\sigma(y_k, \hat{y}) \right]$$

equals, in expectation over the sampled targets, $\text{KL}(G_\theta \parallel \hat{p}_\sigma) + H(G_\theta)$, where $\hat{p}_\sigma = \frac{1}{K} \sum_k k_\sigma(\cdot, y_k)$ is the kernel density estimate of the K data targets, the kernel’s symmetry being what lets the same term be read either way, converging to $\text{KL}(G_\theta \parallel p_\sigma^) + H(G_\theta)$ as $K \rightarrow \infty$. Because each generation is scored on its own, independently of how often the model produces it, L_R is linear in G_θ , so its infimum is approached by collapse onto a point mass, at $K = 1$ the conditional mean, and bare Reverse exploration does not recover p^* . Adding an entropy bonus, that is minimizing $L_R - H(G_\theta)$, cancels the entropy term and leaves the pure reverse KL, minimized as $K \rightarrow \infty$ at $G_\theta = p_\sigma^*$ and reaching p^* as $\sigma \rightarrow 0$.*

Proof sketch. For a fixed draw of targets, $\hat{p}_\sigma$ is a normalized density, so the inner expectation is the cross-entropy between G_θ and $\hat{p}_\sigma$, which decomposes as $\text{KL}(G_\theta \parallel \hat{p}_\sigma) + H(G_\theta)$; by the law of large numbers $\hat{p}_\sigma(\hat{y}) \rightarrow p_\sigma^*(\hat{y})$, giving the limit. The objective weights each generation by a score that does not depend on how often the model produces it, so it is linear in G_θ , and its infimum over densities is approached by densities concentrating toward a point mass at the minimizer of the per-generation cost, the conditional mean at $K = 1$ and a global mode of p_σ^* as K grows. Subtracting $H(G_\theta)$ in the limit leaves the reverse KL, minimized at $G_\theta = p_\sigma^*$. $\square$

Where the blur sits. One further asymmetry is where the blur sits. Forward exploration compares the raw data against the blurred model, whereas Reverse exploration compares the raw model against the blurred data, and as $\sigma \rightarrow 0$ both become comparisons between the raw model and the raw data, in opposite directions.

On the evidence bound. Only the Forward side carries a genuine evidence bound, as the inner $\log \frac{1}{K} \sum_k k_\sigma(y, \hat{y}_k)$ is, in expectation, an importance-weighted lower bound on the model log-likelihood $\log P_{\theta, \sigma}(y)$, as in IWAE [97]. The entropy-corrected Reverse objective $L_R - H(G_\theta)$ is instead a variational free energy for the target p_σ^* as $K \rightarrow \infty$, while bare L_R keeps only its energy term, which is why it collapses and bounds no model likelihood. For this reason we describe Reverse exploration as reverse-KL or variational rather than as an ELBO.

Proposition B.5 (Expressivity of smooth Forward exploration under separation). *Suppose p^* is a mixture of $M^* \leq K$ unimodal components with near-equal masses w_j , each concentrated at scale σ or below, whose modes are pairwise separated by distances $\Delta \gg \sigma$, with $\Delta^2/\sigma^2 \gg (\log K)/\min_j w_j$, and let the assumptions of this subsection hold. Then every minimizer G_θ of the smooth Forward objective places positive mass in a neighborhood of each mode of p^* , so $M(G_\theta) \geq M^*$, and taking $M^* = K$ gives $E \geq K$; at finite K minimizers may also hedge mass between modes, as insurance against candidate draws that miss a mode. Once candidates sufficiently outnumber modes, $K(1 - \min_j w_j)^K \Delta^2/\sigma^2 \lesssim 1$, misses are rare enough that hedging no longer pays: all but a negligible fraction of mass sits at the modes, and $P_{\theta, \sigma}$ has exactly M^* strict local maxima, one per component, or equivalently $M_\sigma(P_\theta) = M^*$.*

Proof sketch. By Proposition B.3, minimizing L_F is maximum likelihood of p^* under the mixture of K candidates drawn i.i.d. from G_θ . If G_θ assigns a mode zero mass, no draw ever covers it, and the objective then pays of order $w_j \Delta^2/\sigma^2$ for a mode of mass w_j , while concentrating that mass elsewhere gains at most $\log K$ in likelihood, plus what it saves by making misses rarer at the remaining modes, with near-equal masses a constant fraction of what the dropped mode costs, so neither term covers the loss and every minimizer covers every mode with positive mass. The same $\log K$ budget forces concentration: spreading mode j 's share to width r costs its covered datapoints of order $w_j r^2/\sigma^2$ against a total possible gain of $\log K$, so under the separation condition each mode's mass sits within $o(\Delta)$ of its center. Each region's peak density therefore dominates the between-mode bridges, whose height is capped by the same budget, and hedge mass, lying at distance $\sim \Delta$ from the peaks at lower density, leaves each peak a strict local maximum, hence $M(G_\theta) \geq M^*$; concentration to $O(\sigma)$ and the exact count require the rare-miss condition. Because candidates are sampled rather than placed, a draw misses a mode of weight w with probability $(1 - w)^K$ at a cost of order Δ^2/σ^2 , so minimizers keep insurance mass *between* modes, which likewise perturbs the optimal mode weights. Under the rare-miss condition that insurance costs more in diluted likelihood than it saves, so all but a negligible fraction of mass concentrates in the mode neighborhoods. Convolving with k_σ then merges each neighborhood's sub- σ structure into a single bump, with exponentially small cross-terms and no spurious maxima between components, so $P_{\theta, \sigma}$ has exactly M^* . $\square$

Assumptions. These statements require (i) the loss to be, up to an additive constant, the negative log of a kernel whose normalizer is independent of $\hat{y}$ and θ (squared error against a Gaussian; for non-normalizable losses such as bounded or perceptual ones the reading fails and these statements do not apply), (ii) the smooth surrogate rather than the hard min, (iii) $K \rightarrow \infty$ only to replace the K -sample mixtures with the blurred densities, and $\sigma \rightarrow 0$ to reach p^* exactly, and (iv) mild regularity: a bounded, symmetric, translation-invariant kernel so that p_σ^* is a density, finite expected loss and differential entropies, domination to justify the limit interchanges, a continuous loss vanishing only at $\hat{y} = y$ for the hard-min limit, and minimizers ranging over all densities. Beyond these, the conventions of Section B.2 carry over unchanged. These statements analyze end-to-end exploration, but because conditioning is suppressed throughout they extend to any single prediction of a factored model whose K candidates share their conditioning and target, with p^* read as that prediction's target conditional, which is what licenses the per-prediction ceiling of Section B.3. For discrete predictions the kernel is the candidate's own softmax, so Proposition B.3's mixture-likelihood reading carries over with no blur, consistent with the atomic convention of Section B.2; the separation-based Proposition B.5 and the kernel-density reading of Proposition B.4 are squared-error statements and do not transfer. The noise-searching hybrids of [1] instead pair each candidate with its own corruption of the datapoint, a coupling search rather than a best-of- K against a fixed target, and we leave their analysis, along with how per-prediction gains compose across a sampler's steps, to future work.

B.6 Why VAEs Break at Scale

Reconstructive training avoids mode blurring by reshaping the objective, but variational autoencoders carry a separate failure that the analysis above does not capture, and it is why they have scaled

worse than diffusion or autoregression. A VAE generates by drawing a latent from a fixed prior $p(z)$ (typically an isotropic Gaussian on $\mathbb{R}^m$) and decoding it, yet during training the decoder only ever sees latents drawn from the *aggregate posterior* $q(z) = \mathbb{E}_{x \sim p_{\text{data}}}[q(z | x)]$ [98]. Sample quality is only as good as the match between these two distributions. The ELBO’s KL term pulls each per-example posterior $q(z | x)$ toward the prior, but it does not force the *aggregate* posterior to fill the prior, so regions of high prior mass can end up with little posterior mass. These are the “prior holes” [99]: latents the decoder is asked to decode at sampling time but never saw during training, which decode to low-quality samples.

This mismatch tends to worsen as the model scales, for two reasons. First, richer data calls for a higher-dimensional latent m , and an isotropic Gaussian in high dimension concentrates its mass on a thin shell whose area grows quickly with m ; a finite training set supplies only N posterior “patches” on that shell, so the fraction of the prior lying near some training latent shrinks as m grows at fixed N . Equivalently, the expected prior mass covered by the aggregate posterior decreases with the latent dimension, so a fresh prior sample is increasingly likely to land in a hole. Second, a more expressive encoder makes this worse rather than better, since it can drive reconstruction error down while letting the aggregate posterior stray further from the prior. Together these explain why VAEs degrade with scale. We make the first reason quantitative below; a complete characterization is left to future work.

A quantitative version. Take the prior $p(z) = \mathcal{N}(0, I_m)$ and suppose, as reconstruction pressure drives per-example posteriors to be narrow [100], that each of the N training examples contributes posterior mass concentrated in a ball of radius $\rho = \sqrt{am}$ around some center μ_n , with $a < 1$ fixed (the prior concentrates at radius $\approx \sqrt{m}$, so $a < 1$ just says each posterior is narrower than the full prior). The Gaussian measure of such a ball is maximized when it is centered at the origin (Anderson’s inequality), where it equals $\Pr[\chi_m^2 \leq am] \leq \exp(-c(a)m)$ with $c(a) = \frac{1}{2}(a - 1 - \ln a) > 0$, a standard chi-square tail bound. A union bound over the N posterior regions then gives

$$\Pr_{z \sim p}[z \text{ lands within } \rho \text{ of some } \mu_n] \leq Ne^{-c(a)m},$$

so a fresh prior sample lands in a hole with probability at least $1 - Ne^{-c(a)m}$: at fixed dataset size N , this approaches one exponentially fast in the latent dimension m , and keeping it constant requires N to grow exponentially with m . The assumptions are deliberately crude (isotropic balls of equal radius, a measure bound in place of actual encoder placements μ_n), and a refined treatment of where the encoder actually places posterior mass is left to future work.

The other end of the dial. The holes above are the weak-penalty failure; Section 8 argues that strengthening the KL penalty only trades them for blurring, and the trade is visible in the ELBO itself. Averaged over the data, the ELBO’s KL term decomposes as

$$\mathbb{E}_{x \sim p_{\text{data}}}[\text{KL}(q(z | x) \| p(z))] = I_q(x; z) + \text{KL}(q(z) \| p(z)),$$

the information the latent carries about the data plus the aggregate mismatch responsible for holes [98]. The penalty prices the two together. Weighting it heavily drives the whole sum down, and since $I_q(x; z)$ is part of the sum, the latent is left carrying little information; an uninformative latent leaves the decoder’s target $p^*(x | z)$ multimodal, at which point the mean-blurring lemma applies to the Gaussian-likelihood decoder and its optimum blurs. Weighting it lightly keeps the latent informative but leaves the mismatch term nearly unpriced, which is exactly what opens holes. As $\bar{M}(p^*)$ grows with data complexity, the information needed to keep per-latent targets near-unimodal grows with it, pulling the two failure modes further apart; no weight avoids both. Diffusion escapes this dial entirely by fixing the encoder (Section 8).