CONDITIONAL FLOW MODELS CONVERT EPISTEMIC UNCERTAINTY INTO APPARENT ALEATORIC VARIANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Conditional Flow Models (CFMs) are increasingly being used in place of deterministic predictors across machine learning, including for tasks that were classically formulated as regression. Given their success in such tasks, we ask the question: *what useful information do the conditional distributions produced by CFMs carry beyond what a point predictor would provide*?

We identify a systematic phenomenon: in many natural regimes, CFM samples routinely have larger variance than the true conditional distribution, even in nearly deterministic regression problems. This effect is most pronounced where the model has not resolved the dependence on the conditioning variable, and thus the additional variance can be viewed as converting *epistemic uncertainty* in fitting the ground truth conditional distribution into apparent *aleatoric variance* in the generated samples. Through controlled synthetic and real-data experiments, and comparisons with explicit smoothing estimators, we show that the induced variance tracks model underfitting of complex or hard-to-fit targets, and decreases with additional training or data. We provide a theoretical account of the phenomenon in a simplified setting and demonstrate a consequent practical use-case of conditional flow-matching: designing high-coverage predictive sets that cover the ground-truth regression target.

1 INTRODUCTION

The role of generative models in machine learning has expanded rapidly. Rather than returning a single point prediction, these models produce a conditional distribution over possible outputs. Perhaps surprisingly, they have gained traction even in settings where predicting a single value, rather than distribution over values, suffices. These setting, which we will call *point-prediction* problems, encompass predicting high-performing actions in robotics (Chi et al., 2023), accurate estimates of a policy value function in reinforcement learning (Agrawalla et al., 2025), and numerous other scientific estimation and time-series problems (Klein et al., 2025).

In this work, we focus on conditional flow-matching models (CFMs), which produce samples by integrating a learned velocity field from base noise to the output space, and suggest that these models confer a complementary benefit: they can ***naturally express estimation uncertainty***, even in point-prediction problem in which observations are nearly *noiseless*.

In the idealized setting, the optimal flow $v^\star$ transports the base distribution to the true conditional law (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023). In practice, however, the flow field is parametrized by a neural network v_θ that is learned imperfectly, due to a combination of limited training data, network capacity, and incomplete optimization (e.g. early stopping, popular in the robot learning community). Thus the learned conditional distribution is shaped not only by the target distribution, but also by the inductive biases caused by fitting the ground-truth vector-field learning with a neural network. In particular, neural networks tend to oversmooth high-frequency features during learning (Rahaman et al., 2019; Xu et al., 2020; Tancik et al., 2020), causing the learned vector-field v_θ to underfit the ground-truth $v^\star$.

We demonstrate, both across domains and in natural theoretical setups, that under this common underfitting behavior, conditional flow models *convert epistemic uncertainty into apparent aleatoric variance*. That is, the approximation error or uncertainty about the dependence of the generated

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

variable on the conditioning variable is expressed as a larger spread in the learned conditional distribution.

Beyond scientific interest, this has actionable consequences. One can train conditional flow-matching models to fit potentially *deterministic target functions* and, as long as the model does not overfit, use the spread of their resulting samples to define confidence sets containing the ground-truth function, with no conformal correction.

Across the application domains we study, including RL policy evaluation, cell perturbation prediction, and robot policy learning, predictive sets built this way are approximately calibrated empirically in the underfitting regime, a property the variance-inflation phenomenon enables. A practitioner receives an uncertainty estimate "for free" alongside the point prediction.

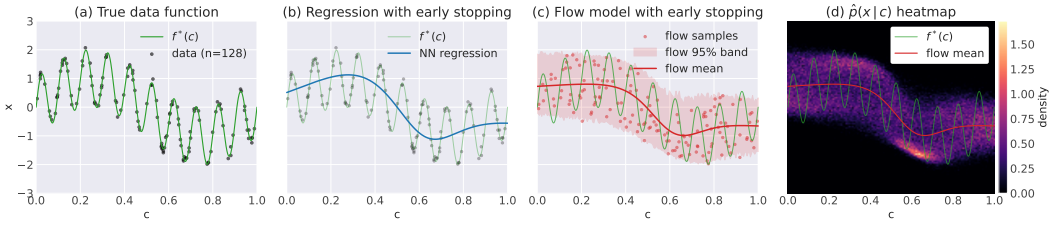


Figure 1: Flow models vs Regression during underfitting.

More concretely, we make the following contributions:

- We identify a **surprisingly general** relationship between learning error in the conditional flow field and the resulting generated samples: whenever a learned CFM underfits the ground-truth velocity field as a function of the conditioning variable, its generated samples have conditional variance strictly larger than that of the true distribution, in expectation over the conditioning variable. This stands in sharp contrast to deterministic regression, where underfitting manifests only as oversmoothing of the predictive mean.
- We trace this variance inflation to a **single mechanism**: smoothing bias in vector-field learning. Any inductive bias that smooths the learned conditional vector field along the conditioning variable, whether early stopping, or explicit regularization, produces the same effect. We demonstrate this empirically on synthetic regression problems, controlled smoothing estimators, and high-dimensional offline RL benchmarks.
- We propose an idealized setting of regularized smoothing to **theoretically study** the variance inflation phenomenon. Under an equivalent-kernel approximation to regularized vector-field learning, the learned conditional distribution is a local mixture of true conditionals around the query point. This yields a clean bias–variance relation: on average, the learned conditional variance is at least the true conditional variance plus the squared bias of the learned conditional mean. This simplified setting serves as a useful explanatory model to understand the variance inflation phenomenon empirically observed in neural CFMs. We also discuss the validity of this approximation to neural CFMs and the regimes where the approximation fails.
- Motivated by our theoretical study, we identify another empirical consequence of the variance inflation phenomenon – the distribution learned by CFMs is approximately calibrated in the underfitting regime. We give a **practical method** for leveraging variance inflation for predictive coverage. Across three application domains, policy evaluation, cell perturbation prediction, and behavior cloning for robotics, predictive sets built directly from CFM samples are approximately calibrated in the underfitting regime, and are competitive with or tighter than conformal-prediction baselines.

2 Related Work

Flow matching and conditional transport. Flow matching and related stochastic-interpolant and rectified-flow methods learn continuous-time transports by regressing a time-dependent vector field along a path between base noise and data (Lipman et al., 2023; Liu et al., 2022; Albergo et al.,

2025). Conditional flow matching indexes this transport by side information, yielding a model of the conditional law of an output given a conditioning variable. Conditional Flow Models (CFMs) have been applied broadly in robotics (Chisari et al., 2024; Rouxel et al., 2024; Zhang et al., 2024) and scientific modeling (Klein et al., 2025; Miller et al., 2024).

Inductive biases and generalization in diffusion and flow models. Several recent works study why diffusion or score-based models generalize rather than simply memorizing the training data. Recent work has attributed this behavior to several sources of bias: variance in the denoising score-matching target (Vastola, 2025), architectural bias (Kamb & Ganguli, 2025), and selective underfitting of the learned score (Song et al., 2025). These papers are closest in spirit to ours, however, they focus on unconditional generation, whereas we study conditional generation and ask how underfitting with respect to the conditioning variable changes the conditional law.

Implicit regularization, spectral bias, and smoothing. Our explanation is based on smoothing bias in the learned conditional vector field. Neural networks are known to preferentially fit low-frequency or smoother structure early in training, a phenomenon often called spectral bias (Rahaman et al., 2019; Xu et al., 2020). In wide-network regimes, training dynamics can be approximated by kernel gradient flow, connecting early stopping and finite-time training to classical regularization in function space (Jacot et al., 2018). Related ideas appear throughout nonparametric regression, smoothing splines, kernel ridge regression, and inverse problems, where regularized estimators can be described by equivalent kernels or linear smoothing operators. We build on this viewpoint, but study a different object: not the bias of a point estimator, but the distribution obtained after integrating a smoothed conditional vector field.

3 Preliminaries

We study conditional generative modeling. Given samples from a joint distribution $p^\star$ over random variables $(X, C) \in \mathbb{R}^{d_x} \times \mathbb{R}^{d_c}$, where X is the generated variable and C is the conditioning variable, conditional generative models learn to generate new samples from the conditional distribution of X given $C = c$, denoted $p^\star(\cdot \mid c)$. Flow models represent such distributions as **continuous normalizing flows** (CNF) and ODEs transporting a simple initial distribution to a terminal distribution approximating $p^\star$. Here, we consider the conditional variant.

Definition 3.1 (Continuous Normalizing Flow). *(Chen et al., 2018) Given a joint distribution p_1 over (X, C), an initial distribution p_0 over (X, C), and conditional vector field $v : [0, 1] \times \mathbb{R}^{d_x} \times \mathbb{R}^{d_x} \to \mathbb{R}^{d_c}$ mapping $(t, x, c) \to v_t(x \mid c)$, we say that (v, p_0) is a **conditional continuous normalization flow (CCNF) for p_1 if, for every $c \in \mathbb{R}^{d_c}$, the random process $(X_t)_{t\in[0,1]}$ defined by the***

$$X_0 \sim p_0(\cdot \mid C = c), \quad \tfrac{\mathrm{d}}{\mathrm{d}t} X_t = v_t(X_t, c) \tag{3.1}$$

satisfies $X_1 \sim p(\cdot \mid c)$. We let $\Phi_t^v(X_0 \mid c)$ denote the deterministic function, called a flow-map, mapping X_0 to X_t in Eq. (3.1). Given (p_0, v), we let $p_t^v(\cdot \mid c, p_0)$ denote the distribution of $X_t \mid c$ marginalized over $X_0 \sim p_0(\cdot \mid c)$ under Eq. (3.1). Note then that, by definition, (v, p_0) is a CCNF for $p_1^v(\cdot \mid \cdot, p_0)$. We suppress dependence on p_0 when clear from context.

Flow-matching (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023; Liu et al., 2022) provides a generic recipe for learning (C)CNFs from data. Let p_0 and differentiable coefficients $\alpha_t, \beta_t : [0, 1] \to \mathbb{R}$ satisfying $\alpha_0 = \beta_1 = 1$ and $\beta_0 = \alpha_1 = 0$ both be given. The induced (linear) **stochastic interpolant** is the random variable $X_t = \alpha_t X_0 + \beta_t X_1$, where $X_0 \sim p_0(\cdot \mid c)$ and $X_1 \sim p^\star(\cdot \mid c)$. Flow-matching is based on the following observation:

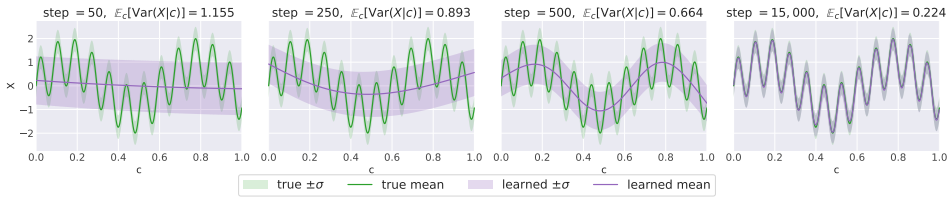
Proposition 3.1 (Albergo et al. (2025)). *Define the vector field $v^\star$ via $v_t^\star(\cdot \mid c) = \mathbb{E}[\dot{X}_t \mid X_t, c]$, where $\dot{X}_t = \dot{\alpha}_t X_0 + \dot{\beta}_t X_1$. Then, $(v^\star, p_0)$ is a CNF for $p^\star$. Moreover, $v^\star$ satisfies $v^\star \in \arg\min_v \mathcal{L}_{\mathrm{FM}}(v)$, where the $\mathcal{L}_{\mathrm{FM}}$ is the flow-matching loss*

$$\mathcal{L}_{\mathrm{FM}}(v) := \mathbb{E}_{(X,C)\sim p^\star} \mathbb{E}_{t\sim \mathrm{Unif}([0,1]), X_0 \sim p_0(\cdot|c)} \mathbb{E}[\|v_t(X_t, C) - \dot{X}_t\|^2] \tag{3.2}$$

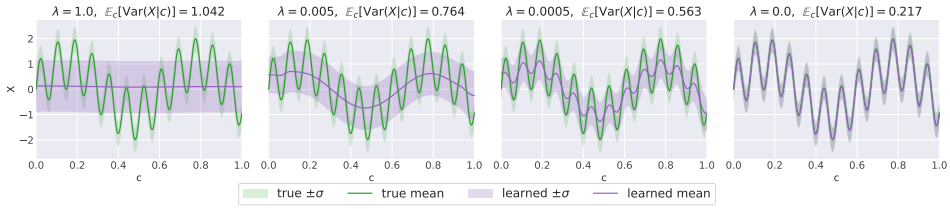
Motivated by Proposition 3.1, flow matching trains a velocity field $v_\theta = (v_{\theta;t}(x, c))$ by (approximately) minimizing an empirical approximation to the flow matching loss $\mathcal{L}_{\mathrm{FM}}$.

4 VARIANCE INFLATION IN UNDERFIT CONDITIONAL FLOWS

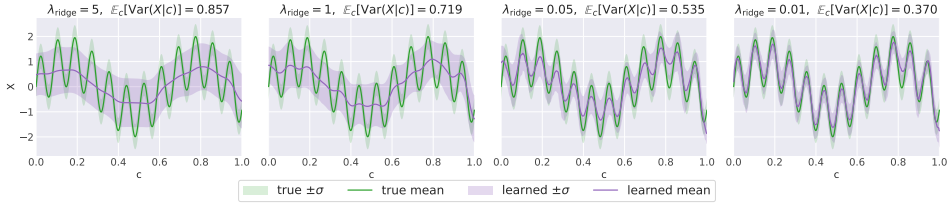
In this section, we uncover two surprising phenomena about underfit conditional flow models: First, the resulting samples have **larger conditional variance** than the ground-truth distribution (in expectation over the conditioning variable), even in nearly deterministic regression problems. Second, this variance inflation is a consequence of **smoothing bias** in the learned vector field: any mechanism that smooths the conditional vector field along the conditioning variable produces the same effect. We demonstrate the phenomenon on a synthetic 1D problem (Figure 1), establish that it is driven by smoothing (Figure 2), and verify that it transfers to a high-dimensional benchmark used in offline RL (Figure 3)



(a) NN vector field parameterization: [Left to Right] Distributions learned with increasing training time



(b) NN vector field parameterization with explicit regularization penalty $\lambda(\partial_c v)^2$



(c) Vector field learning with kernel ridge regression with penalty λ

Figure 2: Any smoothing bias in vector-field learning induces variance inflation. In all three cases, the average learned conditional variance (reported in each panel title) decreases monotonically as smoothing decreases.

Figure 1, given in the introduction, illustrates a one-dimensional synthetic regression problem in which the target function contains both low and high-frequency structure. When a neural-network regressor is trained with early stopping, underfitting leads to oversmoothing of the predictive mean; the learned function captures the coarse trend but fails to recover the finer oscillations.

For a conditional flow model trained under the same regime, the learned conditional mean is similarly oversmoothed, but unlike the deterministic regressor, the flow also learns substantial additional variance, visible in the wide 95% band around the mean. Even in a nearly deterministic problem, underfitting in the conditional vector field appears not only as bias in the learned mean, but also as extra variance in the generated conditional distribution. Panel (d) shows the full learned conditional density $\hat{p}(x \mid c)$, which is broad and roughly bimodal in regions where $f^\star$ (the conditional mean) has high-frequency oscillations within a smoothing neighborhood; the flow's conditional distribution effectively averages over the local oscillations of $f^\star$. This finding is not specific to this example, and Appendix C shows that similar phenomena arise across a number of synthetic target functions.

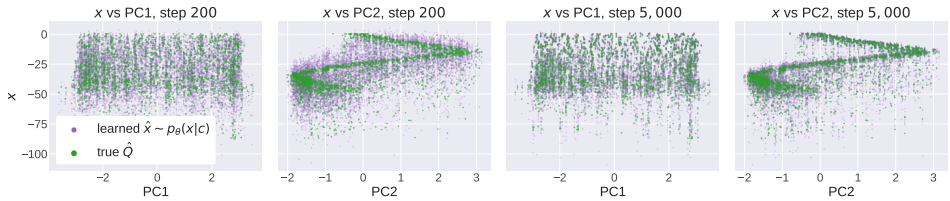
Why do we call this uncertainty epistemic? The excess variance arises from unresolved dependence of the target on the conditioning variable: the learned model smooths over distinctions between nearby conditions that the true conditional mean resolves. This uncertainty is therefore reducible and model-dependent, rather than irreducible randomness in the data-generating process. As the model

resolves more of this structure, for example with additional data or training, both the smoothing bias and the excess variance decrease. However, in a CFM this unresolved variation appears as spread within the learned conditional distribution, and hence as apparent aleatoric variance. From the learned distribution alone, this model-induced spread is not distinguishable from genuine aleatoric variation.

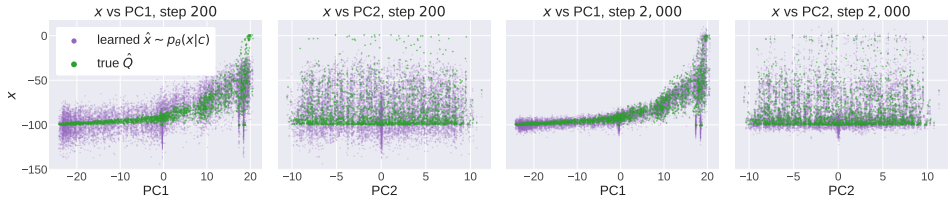
4.1 Variance inflation tracks smoothing bias

Figure 2 reveals that the phenomenon in Figure 1 is not tied to a single estimator or training recipe. We compare three mechanisms that induce smoothing in the learned conditional vector field: early stopping of a neural network which biases training towards low-frequency components of the target (Yao et al., 2007; Rahaman et al., 2019; Tancik et al., 2020), an *explicit roughness penalty* $\lambda\,(\partial_c v)^2$ in the conditioning variable as used in classic non-parametric regression (Wahba, 1990; Green & Silverman, 1994), and kernel ridge regression with an RKHS-norm regularization penalty λ (Caponnetto & De Vito, 2007). All these methods induce smoothing of the conditional mean function, as can be observed from the plotted learned mean at different regularization strengths (Figure 2). In all three cases, this smoothing is accompanied by another consistent effect: the learned conditional variance increases as the learned mean becomes smoother. The average learned conditional variance (reported in each panel title) decreases monotonically as smoothing decreases.

Our key finding, corroborated across additional functions in Appendix C, is that the key object is the smoothing bias in vector-field learning, not flow matching alone. Flow matching simply provides a mechanism through which this smoothing bias is expressed as conditional spread. Also note in Figure 2 that the induced variance is often largest precisely when the learned mean is most biased.



(a) Cube Single



(b) Antmaze

Figure 3: Policy Evaluation: learned (purple) and true (green) conditional distributions against the first two principal components of the state-action conditioning

4.2 Beyond one-dimensional synthetic regression: Monte-Carlo Policy Evaluation

We next test whether the variance-inflation phenomenon persists when the conditioning variable is high-dimensional. For this, we use a Policy Evaluation task, where we train a CFM to predict Q-values conditioned on state-action pairs $c = (s, a)$, using Monte-Carlo targets. We evaluate on two OGBench (Park et al., 2025) environments, `Cube-Single` and `Antmaze-large`, where the conditioning variable is high-dimensional while the target remains scalar.

Figure 3 visualizes the learned and empirical distributions against the first two principal components of the conditioning variable. At early training steps, the learned conditional distribution is noticeably more diffuse than the empirical target, but still 'covers' the target distribution samples; at later checkpoints, the learned samples contract and align more closely with the target scatter. This mirrors the synthetic experiments: underfitting in the conditional vector field leads to excess spread, which decreases as training proceeds.

5 An Idealized Model for Variance Inflation

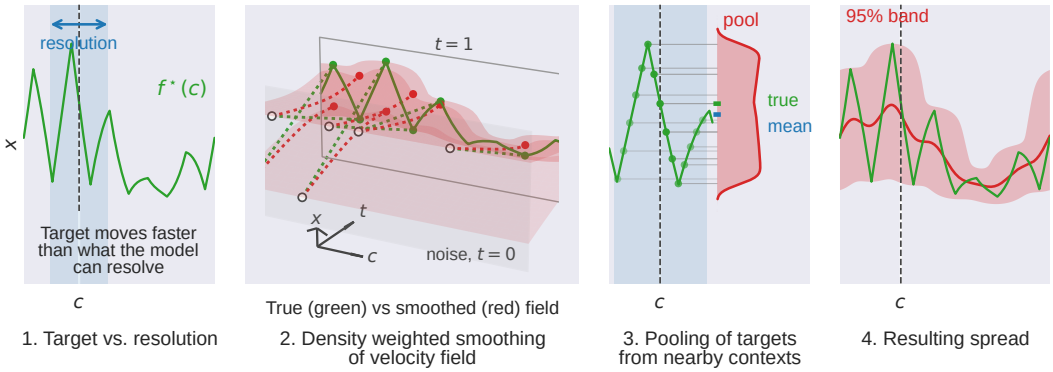


Figure 4: Underfitting of the velocity field leads to local mixing of conditional distributions, not just smoothing of the mean.

Due to finite data, capacity constraints on the learned network, and optimization choices (e.g. early stopping), v_θ is an imperfect approximation of $v^\star$. Because the precise form of this approximation is difficult to characterize for neural function approximators, we study the following idealized population-level **regularized flow-matching problem**.

$$v_\lambda \in \arg\min_{v \in \mathcal{V}} \left\{ \mathbb{E}\left[\left\| v_T(C, X_T) - \dot{X}_T \right\|^2 \right] + \lambda \mathbb{E}\left[\| (L_c v_T)(C, X_T) \|^2 \right] \right\}. \tag{5.1}$$

Above, operator L_c is a linear operator inducing smoothing in the *conditioning variable*. Specifically, for each fixed (t, x), L_c acts on the function $c \mapsto v_t(c, x)$ while treating t and x as fixed. This matches the regime of interest: the learned vector field may depend flexibly on the flow state x and on time t, but its dependence on the conditioning variable c is smoothed. Concrete examples include first-derivative penalties $L_c = \nabla_c$, higher-order smoothness penalties $L_c = \nabla_c^k$, and RKHS-type penalties on $c \mapsto v_t(c, x)$ (Wahba, 1990). It is well-known that such regularized regression problems have equivalent kernel representations (Schölkopf et al., 2001). This objective serves as an idealized model of neural-network smoothing bias; related regularized models have been studied for conditional diffusion models (Pfrommer et al., 2025). Appendix A provides further motivation, and Appendix A.4 numerically evaluates our equivalent-kernel approximation (Equations (5.3) and (5.4)) for neural CFMs on synthetic tasks.

Since L_c only induces smoothing in the conditioning variable c and treats t and x as fixed, for any fixed (t, x), up to a multiplicative constant independent of g, the regularized least-squares problem for the slice $g(c) := v_t(c, x)$ is

$$\hat{g}_{t,x} \in \arg\min_g \left\{ \int_{\mathcal{C}} \left(\|g(c) - v_t^\star(c, x)\|^2 + \lambda \, \|(L_c g)(c)\|^2 \right) p_t^\star(x \mid c) P_C(dc) \right\}. \tag{5.2}$$

For Tikhonov-type penalties, the solution is a linear smoother in the regression target, with some equivalent kernel $K_{\lambda,t,x}$ depending on (t, x) (Silverman, 1984) :

$$v_{\lambda,t}(c, x) = \hat{g}_{t,x}(c) = \int K_{\lambda,t,x}(c, c') \, v_t^\star(c', x) \, p_t^\star(x \mid c') P_C(dc'). \tag{5.3}$$

For the constant-preserving smoothers considered here, the equivalent kernel is normalized so that $\int K_{\lambda,t,x}(c, c') \, p_t^\star(x \mid c') P_C(dc') = 1$. The factor $p_t^\star(x \mid c')$ is the design density of the slice problem, not a separate modeling choice. The starting point of our analysis is a structural assumption on the kernel $K_{\lambda,t,x}$.

Assumption 5.1 (Factorization of the kernel)**.** *The exact kernel $K_{\lambda,t,x}$ depends on (t, x) through the design density $\rho_{t,x}(c) := p_t^\star(x \mid c) p_C(c)$. We adopt the standard equivalent-kernel approximation that assumes the spatial shape of the kernel depends weakly on (t, x):*

$$K_{\lambda,t,x}(c, c') = a_{t,x}(c) \, K'_\lambda(c, c'), \tag{5.4}$$

for some normalization $a_{t,x}(c)$ and a kernel K'_λ that depends on λ but not on (t, x).

Appendix A motivates this assumption and discusses its regime of validity (small t, large path variance, or locally slowly varying conditional mean) in detail. Under Assumption 5.1, together with mild moment and regularity conditions (Assumptions A.1, A.2, and A.3 in Appendix A), the conditional distribution generated by v_λ satisfies the following bias–variance relation.

Theorem 1 (Excess Variance and Covariance)**.** *Suppose Assumption 5.1 and Assumptions A.1, A.2, and A.3 hold. Let $\widehat{X}_1$ denote the endpoint sample generated by integrating v_λ at conditioning C, and let X_1 denote the true target.*

$$\mathbb{E}_C\Big[\mathrm{Cov}(\widehat{X}_1 \mid C)\Big] \succeq \mathbb{E}_C[\mathrm{Cov}(X_1 \mid C)] + \mathbb{E}_C\left[\Big(\mathbb{E}[\widehat{X}_1 \mid C] - \mathbb{E}[X_1 \mid C]\Big)^{\otimes 2}\right]. \tag{5.5}$$

The proof proceeds by showing that Assumption 5.1 reduces the regularized flow-matching solution to a density-weighted local average of the true velocity field, which in turn implies that the endpoint distribution generated by the learned velocity field is a local mixture of the true conditional distributions. Taking moments of this mixture and using standard assumptions on the smoothing kernel (Assumptions A.2, A.3) gives Theorem 1. The full proof is deferred to Appendix A.

The comparison above is in the Loewner order. Equivalently, in every direction u, the average learned conditional variance of $u^\top \widehat{X}_1$ is at least the corresponding true conditional variance plus the average squared directional bias of the learned conditional mean. In the scalar case, Theorem 1 says that, on average over the conditioning variable, the learned conditional variance is at least the true conditional variance plus the squared bias of the learned conditional mean. Thus, any smoothing bias incurred in fitting the conditional mean is reflected as additional spread in the generated samples. Appendix C.2 plots squared bias with excess variance for vector fields trained using different smoothing estimators on multiple one-dimensional functions, and the figure reflects the relation predicted in Eq. (5.5).

6 Coverage Properties of Conditional Flow Models

We now evaluate whether the variance induced by conditional flow models yields meaningful predictive coverage on real-world tasks. The heatmap in panel (d) of Figure 1 suggests that the excess variance of an underfit CFM has structure: it assigns more mass to regions of higher density under the true distribution. Motivated by this, we aim to test a practical consequence of the variance-inflation phenomenon: whether the learned conditional distributions assign appropriate mass to the true outcomes, even in regimes where the conditional mean may be biased. We consider three application domains: Monte-Carlo policy evaluation, cell perturbation prediction, and behavior cloning. We focus on the *underfitting regime*, the regime where the smoothing bias of neural networks holds.

Coverage and interval definitions. For a CFM $\hat{p}_\theta(x \mid c)$ and nominal level $\alpha \in (0,1)$, let $C_{\theta,\alpha}(c)$ denote a predictive set containing approximately α probability mass. We measure empirical coverage on $\mathcal{D} = \{(x_i, c_i)\}_{i=1}^{|\mathcal{D}|}$ as $\widehat{\mathrm{Cov}}_\alpha(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \mathbf{1}\{x_i \in C_{\theta,\alpha}(c_i)\}$. We report coverage at $\alpha \in \{0.75, 0.90, 0.95\}$. For scalar outputs (policy evaluation), we construct $C_{\theta,\alpha}(c)$ as the central α-mass interval of the learned conditional distribution; for high-dimensional outputs (cell-pertubation, behavior cloning), we use sample-based KNN predictive sets (Wang et al., 2022). Full constructions are given in Appendix B.

The question investigated through these experiments is: *Do predictive sets constructed from CFMs as above provide meaningful coverage?* We experiment on the following tasks:

i) Monte-Carlo Policy Evaluation. We use three OGBench environments (Park et al., 2025). The CFM models the distribution of scalar Monte-Carlo returns x conditioned on the state-action pair $c = (s, a)$. Figure 5 reports results for `Antmaze-large`; `Cube-Single` and `Humanoidmaze-medium` are deferred to Appendix B.4.

ii) Single-cell perturbation prediction. We evaluate on Norman, PBMC cytokines, and SciPlex3 from scPerturb (Peidli et al., 2024) datasets, using the CellFlow architecture (Klein et al., 2025). The CFM models the distribution of perturbed-cell states x, with control cells as the source distribution and a pretrained perturbation embedding as the condition c. Figure 6 reports Norman results; PBMC and SciPlex3 are deferred to Appendix B.4.

iii) Behavior cloning. We evaluate a flow-matching policy on PushT (Chi et al., 2023). The CFM models the distribution of short action chunks x conditioned on the robot's observation history c.

Figures 5–7 and 11–14 report our results. We plot CFM training and validation losses, empirical coverage, predictive-set sizes, and utility metrics. We include the loss curves to identify when the trained model begins to overfit. In particular, they indicate the training regime over which we expect coverage to remain reliable: from the beginning of training until the train and validation loss curves diverge (or validation improvement saturates). This is shaded in the figures. In practice, models are generally not trained far into the overfitting regime; techniques such as early stopping, regularization, and large-data training are used to prevent this. Here, however, we deliberately train for longer to demonstrate that coverage remains reliable in the underfitting regime and changes thereafter. For scalar-output tasks we additionally plot pointwise predictive variance against squared prediction error. Set sizes are reported as interval widths for scalar outputs and KNN radii for high-dimensional outputs, while validation log-likelihood (or task success rate for behavior cloning) is tracked to assert that high coverage is not obtained from a degenerate predictive distribution.

Baselines. We compare against six baselines. For scalar outputs, **det + SC** applies split conformal prediction to a deterministic regressor with the same conditioning backbone as the CFM, while **det + CQR** uses conformalized quantile regression to obtain locally adaptive intervals. **Conf CFM** conformalizes the raw CFM predictive sets, whereas **CFM mean + SC** applies split conformal prediction to the CFM conditional mean. We additionally compare against **Deep Evidential Regression (DER)** (Amini et al., 2020) for scalar outputs and **Heteroscedastic Gaussian Regression (HGR)**, using a diagonal Gaussian model for high-dimensional outputs. DER and HGR are reported for fully trained models. Complete experimental details are given in Appendix B. The takeaways are as follows:

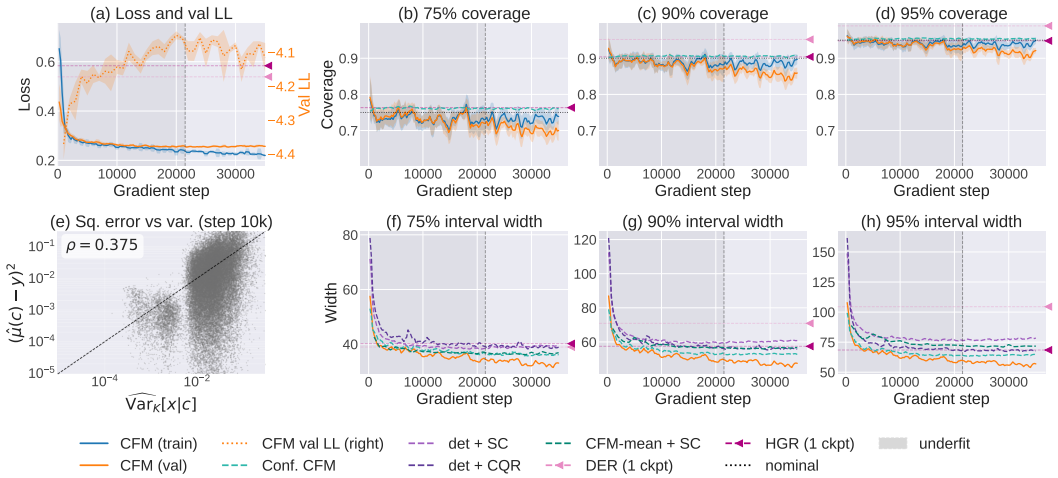


Figure 5: Policy Evaluation: Antmaze-Large

1. **Coverage in the underfitting regime is approximately calibrated.** There are two takeaways here: i) Up till the point CFM train and val loss curves diverge (underfitting regime), coverage of CFM on val data stays approx. $\geq$ nominal level ($\widehat{\mathrm{Cov}}_\alpha(\mathcal{D}) \gtrsim \alpha$), and dips afterwards. ii) Interestingly, coverage of CFMs on the training data stays $\geq$ nominal throughout training. This is because the training data is always 'in-distribution' for the model. Once the model starts to overfit, the training data stays in-distribution w.r.t to the model's learned distribution, whereas the validation does not.
2. **Variance tracks pointwise prediction error.** Examples with larger flow variance also tend to have larger squared prediction error for the flow mean, with positive Spearman correlation across all three environments with one-dimensional outputs (panel (e) in Figures 5, 11, 12). The variance is larger precisely where the model's mean is less accurate, consistent with the interpretation that underfitting is expressed as conditional sample variance.
3. **CFM based coverage sets are tight.** Throughout the regime of interest (before overfitting sets in): i) CFM-based coverage widths / KNN-radii are tighter than split-conformals. This is because split-conformals are not locally adaptive, and also shows that CFM-based coverage sets are. ii) CFM coverage set sizes are competitive with CQR and other baselines that produce close-to nominal

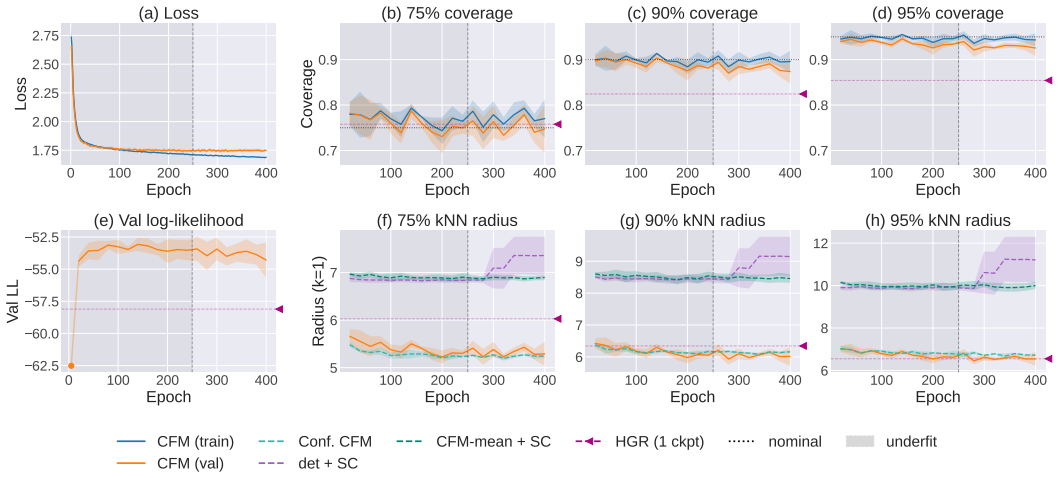


Figure 6: Cell-pertubation dataset: Norman

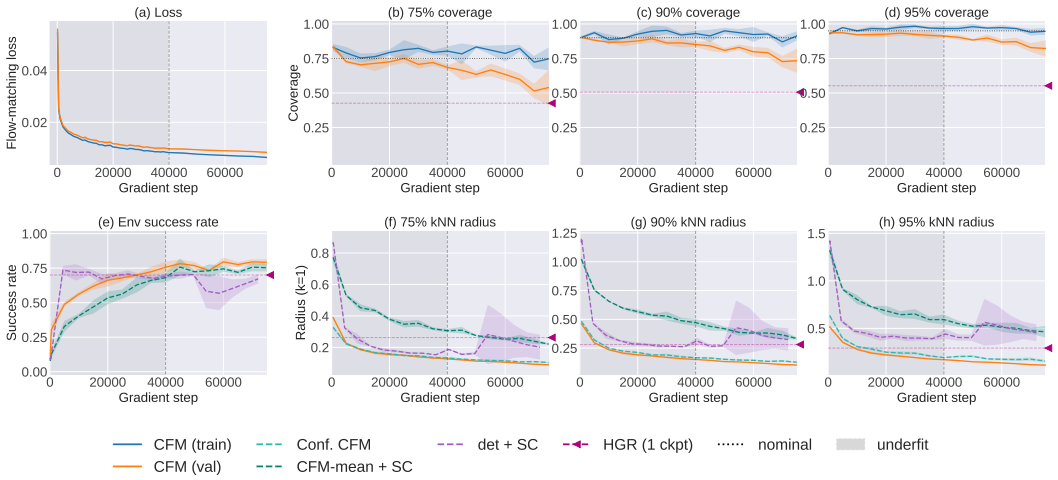


Figure 7: Behavioral Cloning

coverage. iii) CFM and conformalized CFM sets have very similar sizes, showing that additional conformalization changes the sets only minimally.

7 Conclusion and Limitations

We identify a surprising inductive bias of conditional flow models: smoothing bias in vector-field learning causes unresolved dependence on the conditioning variable to appear as excess predictive variance, converting epistemic uncertainty into apparent aleatoric variance. We observe this phenomenon across synthetic and real-world tasks, and provide an idealized model to study it. We empirically find that this excess variance has approximate coverage properties, and, *at any point* in training before overfitting, predictive sets constructed directly from CFM samples achieve approximately nominal coverage without explicit calibration training.

Limitations. Our theory analyzes an idealized population-level regularized smoothing model and does not establish the variance-inflation result for arbitrary neural-network training. Although the theoretical result characterizes the variance inflation phenomenon, it does not imply formal predictive-coverage guarantees. The coverage properties studied here are empirical, and developing stronger theoretical coverage guarantees is left for future work.

AI USE STATEMENT

We used generative AI tools to assist with writing portions of the experimental code, create and refine figures, support literature review including identifying relevant prior work, and assist with minor polishing while writing the manuscript. We did not use AI tools to develop theoretical models or conceptual frameworks, propose hypotheses or interpret results. We have reviewed all AI-assisted material and take responsibility for the final content of this work, including its text, code, figures, scientific claims, and references.

ETHICS STATEMENT

This work investigates the learning behavior and predictive uncertainty of conditional flow models using synthetic data and existing policy-evaluation, cell perturbation, and simulated robotics benchmarks. Applications in safety-sensitive settings would require further validation under the intended deployment conditions.

REPRODUCIBILITY STATEMENT

The appendix provides details of the experimental setups, model architectures, training procedures, predictive-set constructions, and baseline methods, to aid reproducibility. We will also release the code publicly soon to facilitate reproduction of our results.

REFERENCES

Bhavya Agrawalla, Michal Nauman, Khush Agrawal, and Aviral Kumar. floq: Training critics via flow-matching for scaling compute in value-based rl, 2025. URL https://arxiv.org/abs/2509.06863.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023.

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions, 2025. URL https://arxiv.org/abs/2303.08797.

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in neural information processing systems*, 33:14927–14937, 2020.

Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007. doi: 10.1007/s10208-006-0196-8.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Eugenio Chisari, Nick Heppert, Max Argus, Tim Welschehold, Thomas Brox, and Abhinav Valada. Learning robotic manipulation policies from point clouds with conditional flow matching, 2024. URL https://arxiv.org/abs/2409.07343.

Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1):1–49, 2002. doi: 10.1090/S0273-0979-01-00923-5.

Amit Daniely. Sgd learns the conjugate kernel class of the network, 2017. URL https://arxiv.org/abs/1702.08503.

Heinz W Engl, Martin Hanke, and Andreas Neubauer. *Regularization of Inverse Problems*, volume 375 of *Mathematics and Its Applications*. Kluwer Academic Publishers, Dordrecht, 1996.

Peter J Green and Bernard W Silverman. *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman and Hall, London, 1994.

Lukas Heumos, Yuge Ji, Lilly May, et al. Pertpy: an end-to-end framework for perturbation analysis. *bioRxiv*, 2024. doi: 10.1101/2024.08.04.606516.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=ilpL2qACla.

Dominik Klein et al. CellFlow enables generative single-cell phenotype modeling with flow matching. *bioRxiv*, 2025. doi: 10.1101/2025.04.11.648220. Preprint, doi:10.1101/2025.04.11.648220.

Rainer Kress. *Linear Integral Equations*, volume 82 of *Applied Mathematical Sciences*. Springer, New York, 3 edition, 2014. doi: 10.1007/978-1-4614-9593-2.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

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

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022. URL https://arxiv.org/abs/2209.03003.

Benjamin Kurt Miller, Ricky T. Q. Chen, Anuroop Sriram, and Brandon M Wood. FlowMM: Generating materials with riemannian flow matching. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=W4pB7VbzZI.

Thomas M. Norman, Max A. Horlbeck, Joseph M. Replogle, Alex Y. Ge, Albert Xu, Marco Jost, Luke A. Gilbert, and Jonathan S. Weissman. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 365(6455):786–793, 2019. doi: 10.1126/science.aax4438.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl, 2025. URL https://arxiv.org/abs/2410.20092.

Parse Biosciences. 10 Million Human PBMCs in a Single Experiment, 2024. URL https://www.parsebiosciences.com/datasets/10-million-human-pbmcs-in-a-single-experiment/. Accessed: 2026-05-04.

Stefan Peidli, Tessa D Green, Ciyue Shen, Torsten Gross, Joseph Min, Samuele Garda, Bo Yuan, Linus J Schumacher, Jake P Taylor-King, Debora S Marks, Augustin Luna, Nils Blüthgen, and Chris Sander. scPerturb: harmonized single-cell perturbation data. *Nature Methods*, 21:531–540, 2024.

Daniel Pfrommer, Zehao Dou, Christopher Scarvelis, Max Simchowitz, and Ali Jadbabaie. Is your conditional diffusion model actually denoising?, 2025. URL https://arxiv.org/abs/2512.18736.

Zoe Piran, Niv Cohen, Yedid Hoshen, and Mor Nitzan. Disentanglement of single-cell data with biolord. *Nature Biotechnology*, 42:1678–1683, 2024. doi: 10.1038/s41587-023-02079-x.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Quentin Rouxel, Andrea Ferrari, Serena Ivaldi, and Jean-Baptiste Mouret. Flow matching imitation learning for multi-support manipulation, 2024. URL https://arxiv.org/abs/2407.12381.

Christopher Scarvelis, Haitz Sáez de Ocáriz Borde, and Justin Solomon. Closed-form diffusion models. *arXiv preprint arXiv:2310.12395*, 2023.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *Computational Learning Theory*, pp. 416–426. Springer, 2001.

Bernard W Silverman. Spline smoothing: The equivalent variable kernel method. *The Annals of Statistics*, 12(3):898–916, 1984. doi: 10.1214/aos/1176346710.

Kiwhan Song, Jaeyeon Kim, Sitan Chen, Yilun Du, Sham Kakade, and Vincent Sitzmann. Selective underfitting in diffusion models. *arXiv preprint arXiv:2510.01378*, 2025.

Sanjay R. Srivatsan, José L. McFaline-Figueroa, Vijay Ramani, Lauren Saunders, Junyue Cao, Jonathan Packer, Hannah A. Pliner, Daniel L. Jackson, Riza M. Daza, Lena Christiansen, Fan Zhang, Frank Steemers, Jay Shendure, and Cole Trapnell. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 367(6473):45–51, 2020. doi: 10.1126/science.aax6234.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7537–7547, 2020.

John J. Vastola. Generalization through variance: How noise shapes inductive biases in diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=7lUdo8Vuqa.

Grace Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1990. doi: 10.1137/1.9781611970128.

Zhendong Wang, Ruijiang Gao, Mingzhang Yin, Mingyuan Zhou, and David M. Blei. Probabilistic conformal prediction using conditional random samples, 2022. URL https://arxiv.org/abs/2206.06584.

Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *Communications in Computational Physics*, 28(5):1746–1767, 2020.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive approximation*, 26(2):289–315, 2007.

TaeHo Yoon, Joo Young Choi, Sehyun Kwon, and Ernest K Ryu. Diffusion probabilistic models generalize when they fail to memorize. In *ICML 2023 workshop on structured probabilistic inference {\&} generative modeling*, 2023.

Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation, 2024. URL https://arxiv.org/abs/2412.04987.

A THEORETICAL MODEL OF VARIANCE INFLATION

This section provides the formal setup and proof of Theorem 1 from Section 5. The argument proceeds in three steps.

- First, an equivalent-kernel approximation (Section A.1) reduces the regularized flow-matching solution to a density-weighted local average of the true velocity field.
- Second, this representation implies that the endpoint distribution generated by the regularized flow is a local mixture of true conditionals (Proposition A.2).
- Third, taking moments of this mixture gives an explicit decomposition of the learned covariance into true conditional covariance and unresolved variation of the conditional mean. A short operator argument then shows that, on average, this unresolved variation dominates the squared smoothing bias, yielding Theorem 1.

Setup. We restate the regularized flow-matching problem from Section 5: we study the population-level optimum

$$v_\lambda \in \arg\min_{v\in\mathcal{V}} \left\{ \mathbb{E}\left[\|v_T(C,X_T) - \dot{X}_T\|^2\right] + \lambda\, \mathbb{E}\left[\|(L_c v_T)(C,X_T)\|^2\right] \right\}, \tag{A.1}$$

as a proxy for the inductive bias of a learned conditional flow. Here $\mathcal{V}$ is the chosen vector-field class, $\lambda \geq 0$ controls the strength of regularization, and L_c is a linear operator acting only on the conditioning variable: for each fixed (t, x), L_c acts on the function $c \mapsto v_t(c, x)$ while treating t and x as fixed. Examples include derivative penalties such as $L_c = \nabla_c$, higher-order smoothness penalties, and RKHS-type penalties. This captures the regime of interest: the learned vector field may depend flexibly on the flow state x and time t, but its dependence on the conditioning variable c is smoothed.

Justification for studying this setup: Prior work has established that neural networks exhibit a spectral or smoothness bias during training, preferentially fitting smoother and lower-frequency components before finer-scale structure (Daniely, 2017; Jacot et al., 2018; Rahaman et al., 2019; Xu et al., 2020; Lee et al., 2019). Equation (5.1) captures this behavior in an idealized population-level model. Closely related, Pfrommer et al. (2025) study a special case with $L_c = \nabla_c^2$ as a model of smoothness bias in conditional diffusion models. More broadly, underfitting of learned velocity and score fields has been studied extensively in the diffusion and flow-matching literature. Prior work has linked smoothness and selective underfitting of these fields to generalization, interpolation, and hallucination in generative models (Yoon et al., 2023; Scarvelis et al., 2023; **?**; Song et al., 2025). Finally, in Appendix A.4 we numerically evaluate how well the factorized equivalent-kernel representation in Equations (5.3) and (5.4) approximates trained neural CFMs on synthetic one-dimensional regression tasks.

For brevity, throughout this appendix we write $\hat{v} := v_\lambda$ for the regularized population solution.

For any vector field v, the endpoint distribution generated at condition c is the law of $\Phi_1^v(c, Z)$ with $Z \sim \rho_0$. We write its conditional mean and covariance as

$$m_v(c) := \mathbb{E}_Z[\Phi_1^v(c,Z)], \qquad \Sigma_v(c) := \mathrm{Cov}_Z(\Phi_1^v(c,Z)). \tag{A.2}$$

When the generated variable is scalar, we write $s_v^2(c) := \Sigma_v(c)$. In the unregularized realizable population limit, $\Sigma_v(c)$ matches the true conditional covariance of $X_1 \mid C = c$. The central question of this section is what happens to this endpoint covariance when the vector field is regularized or underfit in the c-direction. We show that the learned endpoint distribution can have substantially larger spread than the true conditional distribution, with the excess spread reflecting unresolved variation of the true conditional mean over the model's effective smoothing neighborhood.

Assumption A.1 (Target moments)**.** *The target satisfies*

$$X_1 = \mu^\star(C) + \xi, \qquad \mathbb{E}[\xi \mid C] = 0, \qquad \mathrm{Cov}(\xi \mid C) = \Sigma^\star(C), \tag{A.3}$$

where $X_1 \in \mathbb{R}^{d_x}$ and all second moments are finite. The base variable $X_0 \sim \rho_0$ is independent of (C, ξ), and $p_0^\star(\cdot \mid c) = \rho_0$ does not depend on c. In the scalar case $d_x = 1$, we write $\mu^\star = f^\star$ and $\Sigma^\star = \sigma^2$.

The above assumption simply means that L_2 regression is well-specified and second moments are finite. We make no parametric assumption about ξ beyond its first two conditional moments.

A.1 Assumptions: Equivalent-kernel model of regularized flow matching, and regularity conditions

The regularized objective in (A.1) penalizes variation of $c \mapsto v_t(c, x)$. For each fixed (t, x), this is a supervised regression problem in c whose population design distribution is the conditional law of $C \mid X_t = x$, i.e., proportional to $p_t^\star(x \mid c')P_C(dc')$. Classical equivalent-kernel calculations for regularized least squares show that the fitted value at c is a local average of the regression target (Silverman, 1984), which gives Eq. (5.3). We now simplify this equivalent-kernel representation into the form used in our analysis.

Simplifying the Kernel. We first restate Assumption 5.1 for convenience.

Assumption 5.1 (Factorization of the kernel)**.** *The exact kernel $K_{\lambda,t,x}$ depends on (t, x) through the design density $\rho_{t,x}(c) := p_t^\star(x \mid c)p_C(c)$. We adopt the standard equivalent-kernel approximation that assumes the spatial shape of the kernel depends weakly on (t, x):*

$$K_{\lambda,t,x}(c, c') = a_{t,x}(c)\, K'_\lambda(c, c'), \tag{5.4}$$

for some normalization $a_{t,x}(c)$ and a kernel K'_λ that depends on λ but not on (t, x).

We now show that Assumption 5.1 implies Lemma A.1, and then motivate Assumption 5.1 in the light of prior literature.

Lemma A.1 (Density-weighted equivalent kernel)**.** *For each regularization level λ, there exists a Markov kernel $W_\lambda(dc' \mid c)$ on the conditioning space such that the regularized population velocity in (5.1) satisfies*

$$\hat{v}_t(c, x) \;=\; \frac{\int v_t^\star(c', x)\, p_t^\star(x \mid c')\, W_\lambda(dc' \mid c)}{\int p_t^\star(x \mid c')\, W_\lambda(dc' \mid c)}. \tag{A.4}$$

Proof. Substituting (5.4) into the slice solution Eq. (5.3) and absorbing $a_{t,x}(c)$ into the normalization, we obtain

$$\hat{v}_t(c, x) = \frac{\int K'_\lambda(c, c')\, v_t^\star(c', x)\, p_t^\star(x \mid c')\, p_C(c')\, dc'}{\int K'_\lambda(c, c')\, p_t^\star(x \mid c')\, p_C(c')\, dc'}. \tag{A.5}$$

The form in (A.5) is exactly the representation in Lemma A.1, with W_λ obtained concretely by normalizing $K'_\lambda(c, c')p_C(c')$ in c'. □

Motivating Assumption 5.1. The factorization (5.4) is an equivalent-kernel approximation of the kind used for smoothing splines and other roughness-penalized estimators (Silverman, 1984; Wahba, 1990; Green & Silverman, 1994). The exact kernel $K_{\lambda,t,x}$ depends on (t, x) only through the design density $\rho_{t,x}(c) := p_t^\star(x \mid c)p_C(c)$, so replacing it by a common kernel is justified when $\rho_{t,x}$ is locally stable over the smoothing length scale of the penalty. Equivalently, using the normal-equation resolvent $R_\rho := (M_\rho + \lambda L_c^* L_c)^{-1}$, the second-resolvent identity

$$R_{\rho_{t,x}} - R_{\bar{\rho}} = -R_{\rho_{t,x}}\big(M_{\rho_{t,x}} - M_{\bar{\rho}}\big)R_{\bar{\rho}} \tag{A.6}$$

shows that the exact kernels are stable whenever the perturbation $M_{\rho_{t,x}} - M_{\bar{\rho}}$ is small in the operator norm induced by the regularized problem. This is the standard Fredholm/resolvent picture behind Tikhonov-regularized least squares and kernel ridge regression (Cucker & Smale, 2002; Caponnetto & De Vito, 2007; Kress, 2014; Engl et al., 1996). For the Gaussian-like paths we consider, the approximation is expected to be best for small t, large path variance, or locally slowly varying conditional mean $\mu^\star$. It is weakest near $t = 1$ in low-noise regions, where $p_t^\star(x \mid c)$ sharply localizes around level sets of the conditional mean.

Also, in Section A.4 we numerically test how well Eq. (A.4) approximates the velocity field learned by a neural network, on a synthetic task. This is equivalent to verifying how well Eq. (5.3) and Eq. (5.4) together approximate a neural CFM's learned velocity field.

Remark A.1. *The learned vector field representation Eq. (A.4) obtained from Lemma A.1 is stronger than saying that $\hat{v}_t(c, x)$ is an ordinary pointwise smoother of $v_t^\star(c, x)$: ordinary smoothing in c at fixed (t, x) would replace the conditional mean by a smoothed mean and would not create the same excess endpoint variance. The density-weighted form in (A.5) is structurally different: it locally blends the conditional probability paths themselves, weighting by how likely each c' is to have produced the observed x under the true model. This blending is what produces the variance inflation.*

We conclude this subsection with two assumptions on the resulting kernel W_λ. First, we require mild regularity conditions ensuring that the mixed path above is the path generated by $\hat{v}$.

Assumption A.2 (Path regularity and uniqueness)**.** *For W_λ in Lemma A.1, the mixture density*

$$\hat{p}_t^\lambda(x \mid c) := \int p_t^\star(x \mid c')\, W_\lambda(dc' \mid c) \tag{A.7}$$

is positive and continuously differentiable in (t, x), differentiation under the W_λ-integral is valid, and the multivariate continuity equation

$$\partial_t p_t(x \mid c) + \nabla_x \cdot (p_t(x \mid c)\hat{v}_t(c, x)) = 0 \tag{A.8}$$

with initial law ρ_0 admits a unique weak solution.

Second, we assume reversibility and positive semidefiniteness of the kernel — two standard properties of self-adjoint smoothing operators on $L^2(P_C)$. In particular, for example, they are satisfied by the Markov kernels arising from Tikhonov regularization with symmetric quadratic penalties (Engl et al., 1996).

Assumption A.3 (Symmetric positive smoothing)**.** *The kernel W_λ preserves P_C and is reversible:*

$$P_C(dc)\, W_\lambda(dc' \mid c) = P_C(dc')\, W_\lambda(dc \mid c'). \tag{A.9}$$

Moreover, the smoothing operator

$$(S_\lambda g)(c) := \int g(c')\, W_\lambda(dc' \mid c)$$

is positive semidefinite as an operator on $L^2(P_C)$.

A.2 Formal Main Results

We are now ready to state our formal main results, deferring proofs to Section A.3. First, we show that under our assumptions, **smoothing over the flows directly translates to smoothing over the conditionals**. This is the key bridge between regularization of the vector field and variance inflation in the generated distribution.

Proposition A.2 (Regularization induces a local mixture)**.** *Suppose Assumptions A.1, A.2 and 5.1 hold. Then the conditional path $\hat{p}_t^\lambda(\cdot \mid c)$ defined in Assumption A.2 satisfies the continuity equation*

$$\partial_t \hat{p}_t^\lambda(x \mid c) + \nabla_x \cdot \big(\hat{p}_t^\lambda(x \mid c)\hat{v}_t(c, x)\big) = 0, \tag{A.10}$$

with initial law $\hat{p}_0^\lambda(\cdot \mid c) = \rho_0$. Consequently, the endpoint law generated by $\hat{v}$ is the W_λ-mixture of true conditionals:

$$\hat{p}_1^\lambda(x \mid c) = \int p^\star(x \mid c')\, W_\lambda(dc' \mid c). \tag{A.11}$$

At this point, the flow-dynamics part of the argument is complete: the regularized flow generates a local mixture of the true conditional distributions. The remainder of the variance analysis follows from the moments of this mixture.

Let

$$C'_\lambda \mid C = c \sim W_\lambda(\cdot \mid c), \qquad X'_1 \sim p^\star(\cdot \mid C'_\lambda). \tag{A.12}$$

By Proposition A.2,

$$\hat{X}_1 \mid C = c \overset{d}{=} X'_1. \tag{A.13}$$

Therefore, by the laws of total expectation and total covariance,

$$\hat{\mu}_\lambda(c) := \mathbb{E}[\hat{X}_1 \mid C = c] = \mathbb{E}[\mu^\star(C'_\lambda) \mid C = c] = (S_\lambda \mu^\star)(c), \tag{A.14}$$

$$\begin{aligned}\hat{\Sigma}_\lambda(c) &:= \mathrm{Cov}(\hat{X}_1 \mid C = c) \\ &= \mathbb{E}[\Sigma^\star(C'_\lambda) \mid C = c] + \mathrm{Cov}(\mu^\star(C'_\lambda) \mid C = c).\end{aligned} \tag{A.15}$$

The decomposition in (A.15) already exposes the variance-inflation mechanism. The first term is the true conditional covariance averaged over the smoothing neighborhood, while the second is the variation of nearby conditional means that the smoothed model has failed to resolve.

Variance in the Scalar Case. For simplicity, first consider scalar outputs. Write $\mu^\star = f^\star$ and $\Sigma^\star = \sigma^2$, and define

$$\hat{f}_\lambda(c) := \mathbb{E}[f^\star(C'_\lambda) \mid C = c] = (S_\lambda f^\star)(c), \tag{A.16}$$

$$\delta_\lambda(c) := \hat{f}_\lambda(c) - f^\star(c), \tag{A.17}$$

together with

$$s^2_\lambda(c) := \mathrm{Var}(f^\star(C'_\lambda) \mid C = c), \tag{A.18}$$

$$\bar{\sigma}^2_\lambda(c) := \mathbb{E}[\sigma^2(C'_\lambda) \mid C = c]. \tag{A.19}$$

Specializing (A.14)–(A.15) to the scalar case gives

$$\mathbb{E}[\hat{X}_1 \mid C = c] = f^\star(c) + \delta_\lambda(c), \tag{A.20}$$

$$\mathrm{Var}(\hat{X}_1 \mid C = c) = s^2_\lambda(c) + \bar{\sigma}^2_\lambda(c). \tag{A.21}$$

Equation (A.21) makes the source of the additional variance explicit. The term $s^2_\lambda(c)$ is the variance of $f^\star$ over the smoothing neighborhood $W_\lambda(\cdot \mid c)$, i.e., the unresolved variation of the regression function over the model's effective receptive field around c. The term $\bar{\sigma}^2_\lambda(c)$ is the local average of the irreducible noise variance σ^2 over the same neighborhood. In the unregularized limit ($\lambda \to 0, W_\lambda \to \delta_c$), the first term vanishes and the second collapses to $\sigma^2(c)$, recovering the correct conditional variance.

Thus, the learned conditional variance exceeds the local average noise level by exactly $s^2_\lambda(c)$. The remaining question is whether this unresolved variation is large enough to account for the squared bias $\delta^2_\lambda(c)$ induced by the same smoothing operator.

Proposition A.3 (Average excess variance dominates squared bias)**.** *Suppose* $d_x = 1$. *Under Assumptions A.1 to A.3 and 5.1,*

$$\mathbb{E}_C\left[s^2_\lambda(C) - \delta^2_\lambda(C)\right] = 2\,\langle S_\lambda f^\star, (I - S_\lambda) f^\star \rangle_{L^2(P_C)} \geq 0. \tag{A.22}$$

Average scalar excess variance. Proposition A.3 can be equivalently phrased directly in terms of the conditional variance learned by the flow. Let

$$\hat{V}_\lambda(c) := \mathrm{Var}(\hat{X}_1 \mid C = c) = s^2_\lambda(c) + \bar{\sigma}^2_\lambda(c), \tag{A.23}$$

$$V^\star(c) := \mathrm{Var}(X_1 \mid C = c) = \sigma^2(c), \tag{A.24}$$

and

$$\mathrm{Bias}_\lambda(c) := \mathbb{E}[\hat{X}_1 \mid C = c] - \mathbb{E}[X_1 \mid C = c] = \delta_\lambda(c). \tag{A.25}$$

Combining (A.21) with Proposition A.3 and using $\mathbb{E}[\bar{\sigma}^2_\lambda(C)] = \mathbb{E}[\sigma^2(C)]$ by stationarity,

$$\begin{aligned} &\mathbb{E}_C[\hat{V}_\lambda(C)] - \mathbb{E}_C[V^\star(C)] - \mathbb{E}_C[\mathrm{Bias}_\lambda(C)^2] \\ &\quad = \mathbb{E}_C[s^2_\lambda(C) - \delta^2_\lambda(C)] \geq 0. \end{aligned} \tag{A.26}$$

Equivalently,

$$\mathbb{E}_C[\hat{V}_\lambda(C)] \geq \mathbb{E}_C[V^\star(C)] + \mathbb{E}_C[\mathrm{Bias}_\lambda(C)^2]. \tag{A.27}$$

Thus, on average over the conditioning variable, the learned conditional variance is at least the true conditional variance plus the squared smoothing bias of the learned conditional mean.

This gives a precise sense in which the conditional flow converts epistemic uncertainty into apparent aleatoric variance. The smoothing operator S_λ captures unresolved variation of the regression function $f^\star$ in the conditioning variable, which the model cannot represent given its inductive bias. This unresolved variation manifests as bias $\delta_\lambda(c)$ in the learned conditional mean; (A.27) shows that the same smoothing also induces enough additional conditional spread to cover the squared bias on average. Underfit conditional flows therefore do not merely smooth their predictions toward a less informative estimate; they also produce stochastic outputs whose extra variance carries the unresolved dependence on C.

Vector Case. The vector case follows by applying the same scalar argument along every direction in the output space, and admits the same interpretation.

Proposition A.4 (Vector-valued excess covariance)**.** *Suppose* $X_1 \in \mathbb{R}^{d_x}$. *Let*

$$b_\lambda(c) := \hat{\mu}_\lambda(c) - \mu^\star(c). \tag{A.28}$$

Under Assumptions A.1 to A.3 and 5.1,

$$\mathbb{E}_C[\hat{\Sigma}_\lambda(C)] \succeq \mathbb{E}_C[\Sigma^\star(C)] + \mathbb{E}_C[b_\lambda(C) b_\lambda(C)^\top], \tag{A.29}$$

where $\succeq$ *denotes the Loewner order.*

A.3 Deferred Proofs

We first prove the local-mixture representation that connects smoothing of the velocity field to smoothing of the generated conditional distribution.

Proof of Proposition A.2. For every fixed c', the exact conditional path satisfies the multivariate continuity equation

$$\partial_t p_t^\star(x \mid c') + \nabla_x \cdot (p_t^\star(x \mid c') v_t^\star(c', x)) = 0. \tag{A.30}$$

Integrating this identity against $W_\lambda(dc' \mid c)$ and exchanging the integral with the derivatives, by Assumption A.2, gives

$$\partial_t \hat{p}_t^\lambda(x \mid c) + \nabla_x \cdot \left(\int p_t^\star(x \mid c') v_t^\star(c', x)\, W_\lambda(dc' \mid c) \right) = 0. \tag{A.31}$$

By Lemma A.1,

$$\int p_t^\star(x \mid c') v_t^\star(c', x)\, W_\lambda(dc' \mid c) = \hat{p}_t^\lambda(x \mid c) \hat{v}_t(c, x). \tag{A.32}$$

The initial law is ρ_0 since $p_0^\star(\cdot \mid c') = \rho_0$ does not depend on c'. Uniqueness in Assumption A.2 then identifies $\hat{p}_t^\lambda$ as the law generated by $\hat{v}$. Evaluating at $t = 1$ gives (A.11). □

Proof of Proposition A.3 (scalar case). The proof compares two effects of the same smoothing operator: the unresolved variation s_λ^2 created by averaging nearby values of $f^\star$, and the squared bias δ_λ^2 created by moving $f^\star$ toward $S_\lambda f^\star$. Their difference reduces to a quadratic form involving $S_\lambda(I - S_\lambda)$.

By definition,

$$s_\lambda^2(c) = (S_\lambda f^{\star 2})(c) - (S_\lambda f^\star)(c)^2.$$

Taking expectation over $C \sim P_C$ and using stationarity of W_λ,

$$\mathbb{E}[s_\lambda^2(C)] = \mathbb{E}[f^\star(C)^2] - \mathbb{E}[(S_\lambda f^\star)(C)^2]. \tag{A.33}$$

Similarly, since

$$\delta_\lambda(c) = (S_\lambda f^\star)(c) - f^\star(c),$$

we have

$$\mathbb{E}[\delta_\lambda^2(C)] = \mathbb{E}[(S_\lambda f^\star)(C)^2] - 2\,\mathbb{E}[(S_\lambda f^\star)(C) f^\star(C)] + \mathbb{E}[f^\star(C)^2]. \tag{A.34}$$

Subtracting (A.34) from (A.33) and writing inner products in $L^2(P_C)$ gives

$$\begin{aligned} \mathbb{E}[s_\lambda^2(C) - \delta_\lambda^2(C)] &= 2\langle S_\lambda f^\star, f^\star \rangle - 2\langle S_\lambda f^\star, S_\lambda f^\star \rangle \\ &= 2\langle S_\lambda f^\star, (I - S_\lambda) f^\star \rangle. \end{aligned} \tag{A.35}$$

It remains to show that this quadratic form is nonnegative. By Assumption A.3, S_λ is self-adjoint on $L^2(P_C)$ by reversibility and positive semidefinite by assumption. As a Markov operator preserving P_C, it is also a contraction. Hence its spectrum lies in $[0, 1]$, and so $S_\lambda(I - S_\lambda)$ is positive semidefinite. Therefore,

$$\langle S_\lambda f^\star, (I - S_\lambda) f^\star \rangle = \langle f^\star, S_\lambda (I - S_\lambda) f^\star \rangle \geq 0. \tag{A.36}$$

□

Proof of Proposition A.4 (vector case). The vector result follows by applying the scalar argument to every one-dimensional projection of the conditional mean.

By (A.15),

$$\hat{\Sigma}_\lambda(c) = \mathbb{E}[\Sigma^\star(C'_\lambda) \mid C = c] + A_\lambda(c), \qquad A_\lambda(c) := \mathrm{Cov}(\mu^\star(C'_\lambda) \mid C = c). \tag{A.37}$$

Since W_λ preserves P_C,

$$\mathbb{E}_C[\mathbb{E}[\Sigma^\star(C'_\lambda) \mid C]] = \mathbb{E}_C[\Sigma^\star(C)]. \tag{A.38}$$

It therefore remains to show

$$\mathbb{E}_C[A_\lambda(C)] \succeq \mathbb{E}_C[b_\lambda(C) b_\lambda(C)^\top]. \tag{A.39}$$

Fix $u \in \mathbb{R}^{d_x}$ and define $g_u(c) := u^\top \mu^\star(c)$. Then

$$u^\top A_\lambda(c) u = \mathrm{Var}(g_u(C'_\lambda) \mid C = c), \tag{A.40}$$

$$u^\top b_\lambda(c) = (S_\lambda g_u)(c) - g_u(c). \tag{A.41}$$

Applying the scalar calculation from Proposition A.3 to g_u gives

$$u^\top \big(\mathbb{E}_C[A_\lambda(C)] - \mathbb{E}_C[b_\lambda(C) b_\lambda(C)^\top]\big)\, u = 2\langle S_\lambda g_u, (I - S_\lambda) g_u \rangle_{L^2(P_C)} \geq 0, \tag{A.42}$$

where the final inequality follows from Assumption A.3. Since this holds for every u, the Loewner inequality follows:

$$\mathbb{E}_C[A_\lambda(C)] \succeq \mathbb{E}_C[b_\lambda(C) b_\lambda(C)^\top]. \tag{A.43}$$

Combining this with the covariance decomposition in (A.15) gives (A.29). □

We emphasize that both (A.27) and (A.29) are average statements over C. They do not assert that the scalar excess variance dominates squared bias pointwise, nor that the learned covariance pointwise dominates the true covariance in every heteroscedastic problem. The empirical observations in Section 6 go somewhat beyond this, showing that pointwise predicted variance correlates strongly with pointwise squared error of the flow's mean.

A.4 Validity of the Equivariant-Kernel Assumption (Assumption 5.1)

In this section, we numerically test how well Eq. (A.4) approximates the velocity field learned by a neural network, on a synthetic task. This is equivalent to verifying how well Eq. (5.3) and Eq. (5.4) together approximate a neural CFM's learned velocity field. Let $\hat{v}_\theta(c, x_t, t)$ denote a neural CFM, and v_t^*, p_t^* denote the true velocity field and distribution respectively at time t. Specifically, we want to test if there exists a kernel W such that $\tilde{v}_{W,t}(c, x) = \frac{\int v_t^\star(c', x)\, p_t^\star(x|c')\, W(dc'|c)}{\int p_t^\star(x|c')\, W(dc'|c)}$ closely approximates $\hat{v}_\theta$.

To assess the validity of the equivalent-kernel assumption, we use a 1D Gaussian target:

$$X_1 \mid C = c \sim \mathcal{N}(f^\star(c), \sigma^2), \qquad X_0 \sim \mathcal{N}(0, \sigma_0^2),$$

with linear path $X_t = (1-t)X_0 + tX_1$, giving

$$X_t \mid C = c \sim \mathcal{N}\big(t f^\star(c), s_t^2\big), \qquad s_t^2 = (1-t)^2\sigma_0^2 + t^2\sigma^2.$$

The factorized kernel approximation is most accurate when the slice-design density $\rho_{t,x}(c') = p_t^\star(x \mid c')\, p_C(c')$ varies little over the smoothing bandwidth h_λ. With p_C locally constant, the path mean shifts by $\approx t h_\lambda |f^{\star\prime}(c)|$ over this neighborhood while the path std is s_t, giving the heuristic condition

$$\frac{t h_\lambda |f^{\star\prime}(c)|}{\sqrt{(1-t)^2\sigma_0^2 + t^2\sigma^2}} \ll 1.$$

For our experiments, we use $f^\star(c) = \sin(2\pi c) + \sin(10\pi c)$, $\sigma = 0.05$, and train a CFM $\hat{v}_\theta(c, x_t, t)$, analyzing checkpoints $N_{\text{steps}} \in \{1000, 3000, 5000\}$ (heavy/moderate/mild underfitting).

To test the equivalent-kernel (EK) approximation, we discretize c on a uniform grid $\{c_j\}_{j=1}^N$ and define

$$\tilde{v}_{\phi,t}(c_i, x) = \frac{\sum_j W_{\phi,ij}\, v_t^\star(c_j, x)\, p_t^\star(x \mid c_j)}{\sum_j W_{\phi,ij}\, p_t^\star(x \mid c_j)},$$

where $v_t^\star$ and $p_t^\star$ are available in closed form. We parameterize W_ϕ as a learned-embedding Gibbs kernel,

$$K_{\phi,ij} = \exp\left(-\frac{\|h_\phi(c_i) - h_\phi(c_j)\|^2}{2\sigma_K^2}\right), \qquad W_\phi = D_\phi K_\phi D_\phi,$$

where $h_\phi : [0,1] \to \mathbb{R}^2$ is a small MLP and σ_K is learned. The positive diagonal matrix D_ϕ is obtained by symmetric Sinkhorn normalization so that $W_\phi \mathbf{1} = \mathbf{1}$. This construction ensures

$$W_\phi = W_\phi^\top, \qquad W_\phi \succeq 0, \qquad (W_\phi)_{ij} \geq 0, \qquad W_\phi \mathbf{1} = \mathbf{1}.$$

Denoting $\hat{v}_{\theta,t} = \hat{v}_\theta(\cdot, \cdot, t)$, we fit ϕ by minimizing

$$\mathcal{L}(\phi) = \mathbb{E}_{t,C,X_t \sim \hat{p}_t(\cdot|C)}\left[\left(\tilde{v}_{\phi,t}(C, X_t) - \hat{v}_{\theta,t}(C, X_t)\right)^2\right],$$

where $\hat{p}_t(\cdot \mid c)$ is the marginal induced by integrating the learned field $\hat{v}_\theta$ from the Gaussian base distribution up to time t. Thus, we test whether the EK approximation explains the trained field on the region of state space actually visited by the learned model.

Let $\hat{p}_1(\cdot \mid c), \tilde{p}_1(\cdot \mid c)$ be the endpoint distributions obtained by integrating $\hat{v}_\theta, \tilde{v}_\phi$ from a shared $\mathcal{N}(0,1)$ base, with means/variances $\hat{f}, \hat{\sigma}^2$ and $\tilde{f}, \tilde{\sigma}^2$ respectively. Writing

$$\langle \cdot \rangle = \mathbb{E}_{C,X_t \sim \hat{p}_t(\cdot|C)}[\cdot],$$

we report the field error

$$E_{\text{field}}(t) = \frac{\left\langle (\tilde{v}_{\phi,t} - \hat{v}_{\theta,t})^2 \right\rangle}{\left\langle \hat{v}_{\theta,t}^2 \right\rangle}$$

as a function of t in Figure 8, Column 1. We also plot $\tilde{v}_{\phi,t}(c, x_t)$ and $\hat{v}_{\theta,t}(c, x_t)$ as a function of c at $x_t = 0$ for $t = 0.25, 0.5, 0.7, 0.9$ in Figure 8, Columns 2–5. As expected, Figure 8 shows that $\tilde{v}_\phi$ is a good approximation of $\hat{v}_\theta$ over much of the learned trajectory, with the approximation becoming weaker in some regions as $t \to 1$. In Figure 10, we also plot the endpoint distributions obtained by integrating $\hat{v}_\theta$ and $\tilde{v}_\phi$. Interestingly, the endpoint distributions look very similar, and the vector-field approximation error close to $t = 1$ does not affect the integral significantly. Finally, in Figure 9 we visualize how both $\hat{v}_\theta$ and $\tilde{v}_\phi$ look like density-weighted smoothed functions of the true velocity field $v^\star$.

B Coverage Experiments

This section provides full constructions of the predictive sets and conformal baselines used in Section 6, along with experimental details.

B.1 Coverage Set Creation

Flow predictive intervals (scalar outputs) In the Monte-Carlo policy-evaluation experiments, the target variable is one-dimensional. We write each example as (x_i, c_i), where c_i is the conditioning input and $x_i \in \mathbb{R}$ is the target value. Given a trained conditional flow with flow map $\Phi_1^\theta(\cdot \mid c)$ and base distribution ρ_0, the learned conditional distribution at condition c is the law of

$$\Phi_1^\theta(Z \mid c), \qquad Z \sim \rho_0.$$

Because the output is one-dimensional, flow trajectories do not cross. The map $z \mapsto \Phi_1^\theta(z \mid c)$ is therefore order-preserving, and central predictive intervals can be obtained by transporting quantiles of the base distribution. For a nominal coverage level $\alpha \in (0,1)$, let

$$z_\alpha^- = \text{Quantile}_{(1-\alpha)/2}(\rho_0), \qquad z_\alpha^+ = \text{Quantile}_{(1+\alpha)/2}(\rho_0).$$

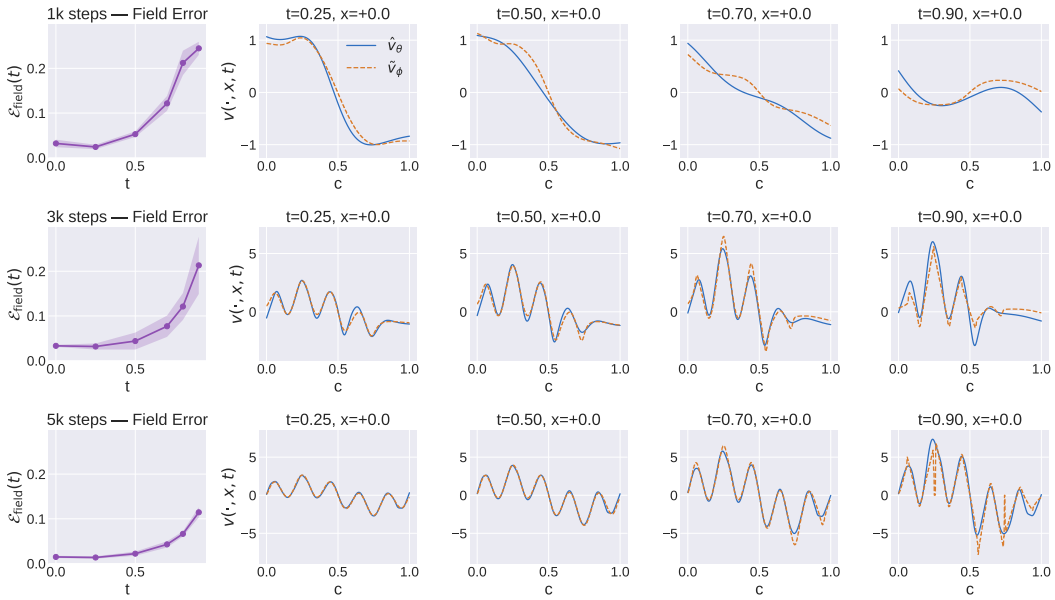


Figure 8: Kernel approximation error and velocity field visualization

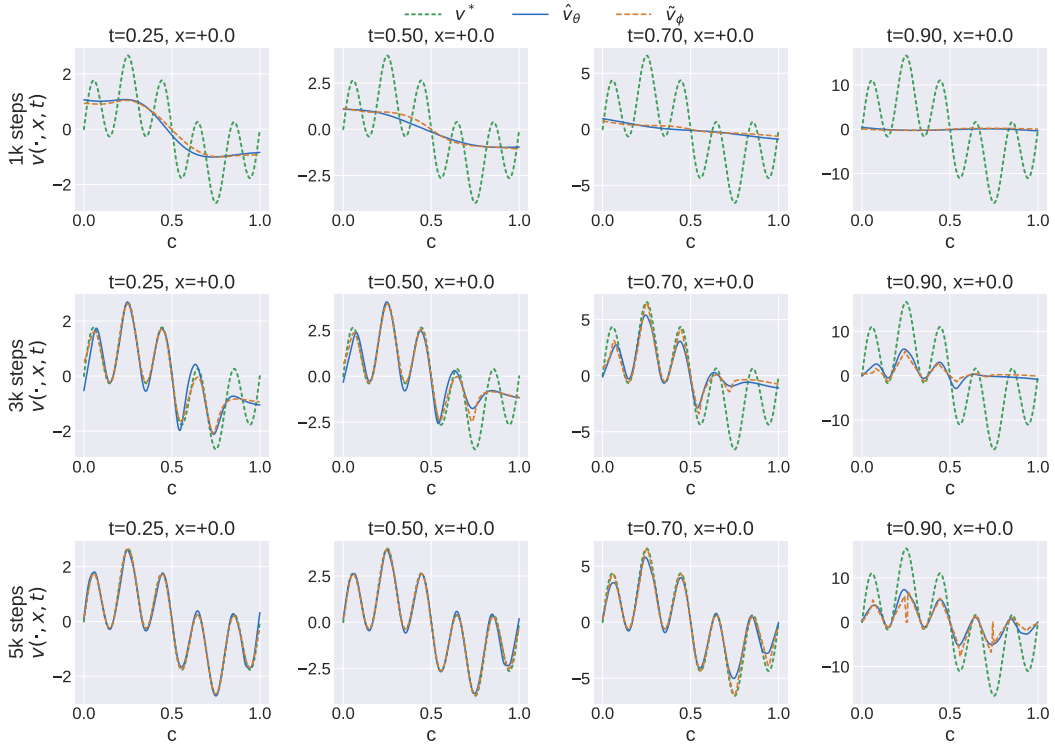


Figure 9: Visualization of density-weighted smoothing in the learned velocity field and the kernel approximation

We define the raw flow predictive interval by

$$C^{\text{flow}}_{\theta,\alpha}(c) = \left[\Phi^\theta_1(z^-_\alpha \mid c),\ \Phi^\theta_1(z^+_\alpha \mid c)\right]. \tag{B.1}$$

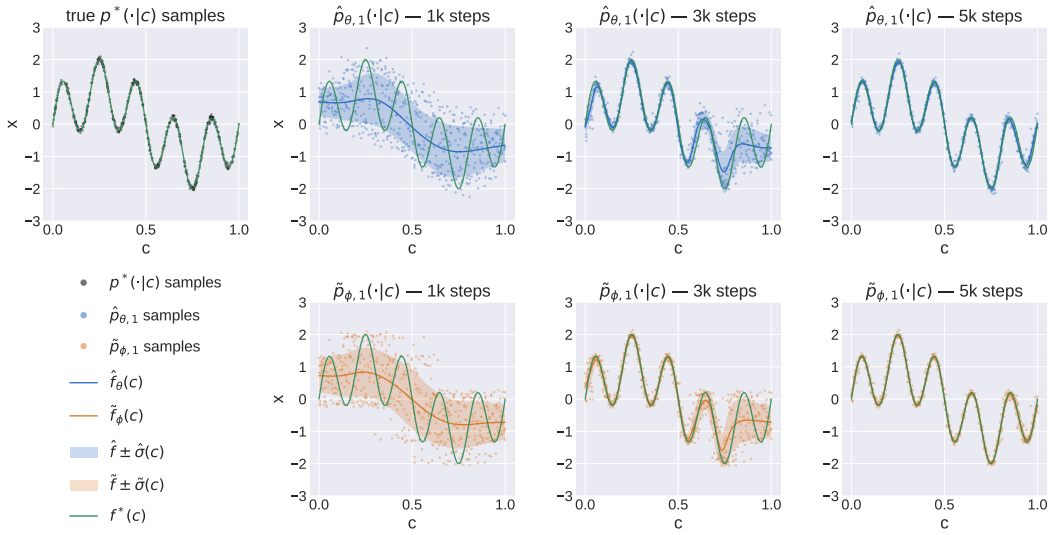


Figure 10: Effect of kernel approximation on the endpoint distribution

Equivalently, this is the central α-mass interval of the learned one-dimensional conditional distribution. The corresponding interval width is

$$W_\alpha^{\text{flow}}(c) = \Phi_1^\theta(z_\alpha^+ \mid c) - \Phi_1^\theta(z_\alpha^- \mid c). \tag{B.2}$$

For a dataset $\mathcal{D}$, we report the empirical coverage $\widehat{\text{Cov}}_\alpha(\mathcal{D})$ and the average interval width

$$\overline{W}_\alpha^{\text{flow}}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x_i,c_i)\in\mathcal{D}} W_\alpha^{\text{flow}}(c_i). \tag{B.3}$$

KNN predictive sets (high-dimensional outputs) For high-dimensional outputs, central quantile intervals are not well-defined, and the standard conformal baselines (split conformal, CQR) do not generalize naturally to multivariate outputs. We instead use a sample-based KNN predictive set, similar to what is proposed in Wang et al. (2022).

For a condition c, draw $\tilde{x}_1, \ldots, \tilde{x}_M \sim \hat{p}_\theta(\cdot \mid c)$. For any candidate point x, define $d_k(x; c)$ as the distance from x to its k-th nearest neighbor among the generated samples. We compute the same score for the generated samples themselves using leave-one-out nearest neighbors, and define

$$\tau_{\theta,\alpha}(c) = \text{Quantile}_\alpha\left(\{d_k(\tilde{x}_m; c)\}_{m=1}^M\right).$$

The KNN predictive set is

$$C_{\theta,\alpha}^{\text{KNN}}(c) = \{x : d_k(x; c) \le \tau_{\theta,\alpha}(c)\}. \tag{B.4}$$

A held-out target x_i is counted as covered if $d_k(x_i; c_i) \le \tau_{\theta,\alpha}(c_i)$. Throughout, we use $k = 1$ for computing coverage metrics.

B.2 Baselines

All conformal methods use a held-out calibration set $\mathcal{D}_{\text{cal}}$ disjoint from the training set. The deterministic baselines use the same backbone architecture as the flow's conditioning encoder and are trained on the same training set with matched optimizer, learning rate, and number of gradient steps. Split conformal prediction is applied either to a deterministic regressor (**det + SC**) or to the conditional mean of the trained CFM (**CFM mean + SC**). **Conf CFM** instead applies conformal calibration directly to the raw predictive set produced by the CFM.

Split conformal prediction. Let $\hat{f}(c)$ denote a point predictor. For **det + SC**, $\hat{f}(c)$ is the output of the deterministic regressor, whereas for **CFM mean + SC**, $\hat{f}(c)$ is the conditional mean of the trained CFM.

On $\mathcal{D}_{\text{cal}}$, we compute absolute residual scores

$$r_i = |x_i - \hat{f}(c_i)|.$$

Let q_α^{split} be the α-quantile of $\{r_i\}$, with the standard finite-sample correction. The split-conformal interval is

$$C_\alpha^{\text{split}}(c) = \left[\hat{f}(c) - q_\alpha^{\text{split}},\ \hat{f}(c) + q_\alpha^{\text{split}}\right],$$

with width $W_\alpha^{\text{split}}(c) = 2q_\alpha^{\text{split}}$. Every input receives the same expansion, so split conformal achieves marginal coverage but is not locally adaptive.

For high-dimensional outputs, **CFM mean + SC** uses the analogous construction with Euclidean residual scores

$$r_i = \left\|x_i - \hat{f}(c_i)\right\|_2.$$

The resulting predictive set is the conformal ball

$$C_\alpha^{\text{CFM-mean+SC}}(c) = \left\{x : \left\|x - \hat{f}(c)\right\|_2 \le q_\alpha^{\text{split}}\right\}.$$

Conformalized quantile regression. For **det + CQR**, at each coverage level α, we train two predictors $\hat{\ell}_\alpha(c)$ and $\hat{u}_\alpha(c)$ using quantile losses for the lower and upper tails, with target quantile levels $(1-\alpha)/2$ and $(1+\alpha)/2$, respectively. Before conformal calibration, these define the interval $[\hat{\ell}_\alpha(c), \hat{u}_\alpha(c)]$.

On $\mathcal{D}_{\text{cal}}$, we compute CQR scores

$$s_i = \max\left\{\hat{\ell}_\alpha(c_i) - x_i,\ x_i - \hat{u}_\alpha(c_i)\right\}.$$

Let q_α^{CQR} be the α-quantile of $\{s_i\}$, with the standard finite-sample correction. The conformalized CQR interval is

$$C_\alpha^{\text{CQR}}(c) = \left[\hat{\ell}_\alpha(c) - q_\alpha^{\text{CQR}},\ \hat{u}_\alpha(c) + q_\alpha^{\text{CQR}}\right],$$

with width

$$W_\alpha^{\text{CQR}}(c) = \hat{u}_\alpha(c) - \hat{\ell}_\alpha(c) + 2q_\alpha^{\text{CQR}}.$$

Unlike split conformal prediction, CQR can produce different interval widths at different conditioning inputs.

Conformalized CFM. For scalar outputs, let

$$C_{\theta,\alpha}(c) = \left[\hat{\ell}_{\theta,\alpha}(c), \hat{u}_{\theta,\alpha}(c)\right]$$

denote the raw CFM predictive interval. On $\mathcal{D}_{\text{cal}}$, we compute the conformity scores

$$s_i^{\text{CFM}} = \max\left\{\hat{\ell}_{\theta,\alpha}(c_i) - x_i,\ x_i - \hat{u}_{\theta,\alpha}(c_i)\right\}.$$

Let q_α^{CFM} be their α-quantile, with the standard finite-sample correction. The conformalized CFM interval is

$$C_{\theta,\alpha}^{\text{ConfCFM}}(c) = \left[\hat{\ell}_{\theta,\alpha}(c) - q_\alpha^{\text{CFM}},\ \hat{u}_{\theta,\alpha}(c) + q_\alpha^{\text{CFM}}\right].$$

For high-dimensional outputs, we conformalize the KNN predictive set. Let $\tau_{\theta,\alpha}(c)$ denote the raw CFM KNN threshold defined above. We compute calibration scores

$$s_i^{\text{CFM}} = d_k(x_i; c_i) - \tau_{\theta,\alpha}(c_i)$$

and let q_α^{CFM} be their conformal quantile. The resulting predictive set is

$$C_{\theta,\alpha}^{\text{ConfCFM}}(c) = \left\{x : d_k(x; c) \le \tau_{\theta,\alpha}(c) + q_\alpha^{\text{CFM}}\right\}.$$

Thus, Conf CFM applies a conformal correction to the boundary of the raw CFM predictive set without changing the underlying learned distribution.

Deep Evidential Regression. We use the standard Deep Evidential Regression method of Amini et al. (2020) as an additional probabilistic uncertainty baseline. DER is evaluated only for scalar outputs, and its predictive intervals are obtained from its learned predictive distribution. We report DER at its best validation checkpoint.

Heteroscedastic Gaussian Regression. Heteroscedastic Gaussian Regression models the conditional distribution as

$$p_\psi(x \mid c) = \mathcal{N}\left(x; \mu_\psi(c), \Sigma_\psi(c)\right).$$

For scalar outputs, the network predicts a conditional mean $\mu_\psi(c)$ and variance $\sigma^2_\psi(c) > 0$, and is trained by minimizing the Gaussian negative log-likelihood

$$\mathcal{L}_{\mathrm{HGR}} = \frac{1}{2}\left[\frac{\left(x - \mu_\psi(c)\right)^2}{\sigma^2_\psi(c)} + \log \sigma^2_\psi(c)\right],$$

up to an additive constant. The central α-mass predictive interval is

$$C^{\mathrm{HGR}}_{\psi,\alpha}(c) = \left[\mu_\psi(c) + z_{(1-\alpha)/2}\sigma_\psi(c),\ \mu_\psi(c) + z_{(1+\alpha)/2}\sigma_\psi(c)\right],$$

where z_q denotes the q-quantile of the standard normal distribution.

For d-dimensional outputs, HGR predicts a diagonal conditional covariance,

$$\Sigma_\psi(c) = \mathrm{diag}\left(\sigma^2_{\psi,1}(c), \ldots, \sigma^2_{\psi,d}(c)\right),$$

and minimizes

$$\mathcal{L}_{\mathrm{HGR}} = \frac{1}{2}\sum_{j=1}^{d}\left[\frac{\left(x_j - \mu_{\psi,j}(c)\right)^2}{\sigma^2_{\psi,j}(c)} + \log \sigma^2_{\psi,j}(c)\right].$$

Computing k-NN radii across methods. For high-dimensional outputs, we express all methods' predictive-region sizes on the same k-NN radius scale used for the CFM. At each condition c, the sample-based radius is the α-quantile of the leave-one-out k-nearest-neighbor distances among K samples from the predictive distribution; we report its average over conditions. We hold K and k fixed across methods within each task to ensure a common sampling resolution. This preserves the CFM's original radius statistic while avoiding an ellipsoidal approximation to its predictive region.

For the CFM, we sample from the learned conditional flow; for HGR, from the predicted diagonal Gaussian. HGR radii are computed separately at each condition because Euclidean k-NN distances depend on the full shape of the diagonal covariance, not only on the geometric mean of its marginal standard deviations.

For the point-predictor split-conformal baselines (**det + SC** and **CFM mean + SC**), no per-condition sampling is required. At each nominal level α, we interpret the conformal L_2 ball of radius R_α as the central α-mass region of an isotropic Gaussian $\mathcal{N}(\mu(c), s^2_\alpha I_D)$, where $\mu(c)$ is the corresponding point prediction and D is the output dimension. This fixes

$$s_\alpha = \frac{R_\alpha}{\sqrt{\chi^2_\alpha(D)}},$$

where $\chi^2_\alpha(D)$ denotes the α-quantile of the chi-squared distribution with D degrees of freedom. Since pairwise distances are translation-invariant and scale linearly with s_α, the corresponding radius is

$$\tau_\alpha = \frac{R_\alpha}{\sqrt{\chi^2_\alpha(D)}}\, q_{\alpha,k}(K, D),$$

where $q_{\alpha,k}(K, D)$ is the reference radius obtained by applying the same leave-one-out k-NN quantile procedure to a cloud of K standard Gaussian samples in D dimensions. We estimate this reference quantity once by simulation for each (α, k, K, D) and reuse it for both ball-based baselines. The distance rescaling is exact under the isotropic-Gaussian representation; only the reference quantity is estimated numerically. This representation is used solely to express the ball radius on the common k-NN scale.

For **Conf CFM**, we report the sample-based CFM threshold after adding the conformal correction defined above, again averaged over conditions.

B.3 Datasets and Training Setup

All the plots are created using 3 random seeds. Dataset specific details and training setup are given below.

B.3.1 Biology datasets

We evaluate conditional flow matching on three single-cell perturbation benchmarks from the harmonized scPerturb collection (Peidli et al., 2024): **Norman**, **SciPlex3**, and **PBMC cytokines**. In each dataset, a cell is represented by a PCA-reduced gene-expression vector $x \in \mathbb{R}^d$, and each sample is annotated with a perturbation condition c together with a control condition. The perturbation prediction problem is cast as conditional generation: the source distribution is the population of control cells, the target distribution is the population of cells under the perturbation condition, and the conditioning variable c is a pretrained embedding of the perturbation. With the source distribution of control cells fixed, the flow learns the distribution of perturbed-cell states conditioned on c.

Norman (Perturb-seq). The Norman et al. (Norman et al., 2019) Perturb-seq dataset consists of single-cell RNA-seq profiles of K562 cells subjected to CRISPRa-induced overexpression of one or two genes. We use the preprocessed AnnData release distributed by the authors of biolord (Piran et al., 2024), which contains ∼89k cells across 229 single and double-gene perturbation conditions plus unperturbed controls; this is the same release used by the CellFlow benchmark (Klein et al., 2025). Of biolord's predefined train/test partitions, we use the "train" perturbation conditions (∼42k cells) for our flow-matching experiments and the in-distribution cells are then randomly split into a training set and a validation set (80-20 split). The 7353 unperturbed control cells form the pool from which x_0 samples are drawn at training time. The condition $c \in \mathbb{R}^{2560}$ is the concatenation of two ESM2-650M (Lin et al., 2023) protein embeddings, one per perturbed gene. The target $x \in \mathbb{R}^{50}$ is the 50-component PCA projection of the log-normalized expression vector, with PCA fit on all cells.

The x_0 distribution is control-pool + Gaussian noise: $x_0 = z + \sigma\,\epsilon$ with $z \sim \mathrm{Uniform}(\text{control pool})$, $\epsilon \sim \mathcal{N}(0, I)$, and $\sigma = 1.0$. The velocity field is a CellFlow-style (Klein et al., 2025) three-branch MLP: a condition encoder ($[1024, 1024] \to 256$), a cell-state encoder ($[512, 512, 512] \to 512$), and a sinusoidal time embedding (128d) followed by $[512, 512, 512] \to 512$, fused by a $[1024, 1024, 1024] \to 50$ decoder. Activations are SiLU throughout. We train with Adam at learning rate 10^{-3}, batch size 1024. Sampling at evaluation uses 100 Euler steps.

PBMC (cytokine stimulation). The PBMC cytokine atlas released by Parse Biosciences (Parse Biosciences, 2024) contains single-cell RNA-seq profiles of peripheral blood mononuclear cells from 12 donors stimulated with 90 cytokines plus PBS controls (∼9.7M cells over 2,000 highly-variable genes). We access this dataset through the CellFlow data loader (Klein et al., 2025) and restrict to a single donor (Donor 1), which contains ∼1.3m cells: ∼90k PBS-stimulated controls and ∼1.2m cytokine-stimulated cells across the 90 cytokines. To keep the per-condition cell count balanced, we subsample uniformly at random to at most 1,000 cells per cytokine condition (and the same cap for controls), yielding 91,000 cells which are split into a training set and a validation set (80–20 split). The 1000 subsampled PBS controls form the pool from which x_0 samples are drawn at training time. The condition $c \in \mathbb{R}^{2560}$ is the ESM2 (Lin et al., 2023) protein embedding of the cytokine. The target $x \in \mathbb{R}^{100}$ is the 100-component PCA projection of the log-normalized expression vector, with PCA fit on all cells.

The x_0 distribution is again control-pool + Gaussian noise with standard deviation $\sigma = 10.0$. The velocity field used a condition encoder ($[1024, 1024] \to 256$), a cell-state encoder ($[512, 512, 512] \to 512$), and a sinusoidal time embedding (1024d) followed by $[1024, 1024, 1024] \to 1024$, fused by a $[1024, 1024, 1024] \to 100$ decoder. We use SiLU activations, train with Adam at learning rate 10^{-3}, batch size 1024. Sampling at evaluation uses 1,000 Euler steps.

SciPlex (small-molecule perturbations). The SciPlex3 dataset of Srivatsan et al. (Srivatsan et al., 2020) profiles single-cell transcriptomes of three cancer cell lines (A549, MCF7, K562) exposed to ∼188 small-molecule drugs at four doses each, with DMSO as the vehicle control. We restrict to the A549 lung-adenocarcinoma line, retrieve the dataset through the `pertpy` library (Heumos et al., 2024) via `pertpy.data.sciplex3_raw()` and follow the preprocessing of the CellFlow benchmark (Klein et al., 2025). After filtering, the dataset contains

703 in-distribution conditions covering 187 drugs, with ∼136k perturbed cells and ∼3k DMSO control cells, which are split into train-val datasets (80-20 split). The condition $c \in \mathbb{R}^{1025}$ is [Morgan-1024-fingerprint(drug), $\log_{10}$(dose in nM)]. The target $x \in \mathbb{R}^{100}$ is the 100-component PCA projection of the log-normalized expression vector, with PCA fit on all cells.

Similar to the previous two datasets, th x_0 distribution is constructed by adding gaussian noise with standard-deviation $= 10.0$ on the control pool. The velocity field architecture is the same as that used in the Norman experiments. Activations are SiLU throughout. We train with Adam at learning rate 10^{-3}, batch size 1024, for 1,000 epochs. Sampling at evaluation uses 1000 Euler steps.

B.3.2 Policy Evaluation

We construct three policy-evaluation datasets on top of the OGBench (Park et al., 2025) offline RL benchmark, one per environment. For each environment, we take the trained floq agent of Agrawalla et al. (2025) as the policy π to be evaluated, and distill its true return $Q^\pi(s, a)$ into a flow-matching predictor by Monte-Carlo: for each (s, a) pair we average the discounted return over n_{MC} independent on-policy rollouts of π from (s, a), giving a stochastic target $\hat{Q}(s, a) \in \mathbb{R}$. The condition is $c = [s, a]$ and the target is $x = \hat{Q}(s, a)$.

The per-environment details are:

Dataset	n_{MC}	H	γ	$\dim s$	$\dim a$	$\dim c$	$\lvert\mathcal{D}\rvert$
Cube-single	20	200	0.99	28	5	33	200,000
AntMaze-large	20	1000	0.998	29	8	37	200,000
Humanoid-Maze-medium	20	2000	0.999	69	21	90	200,000

Table 1: n_{MC} is the number of independent on-policy rollouts averaged into $\hat{Q}(s, a)$, H is the fixed rollout horizon, and γ is the discount factor.

x_0 distribution. All three FloQ runs use an isotropic Gaussian prior $x_0 \sim \mathcal{N}(0, 1)$ in the 1-D target space.

Architecture and training. For all three FloQ datasets we use the same MLP velocity field with hidden widths $[512, 512, 512, 512]$ and ReLU activations, taking the full input $[c, x_t, \mathrm{TimeEmb}(t)]$, where TimeEmb is a 32-dimensional cosine-geometric time embedding. The output is a scalar velocity. We optimize with Adam (lr 3×10^{-4}, no weight decay) using batch size 1024.

B.3.3 Behavior Cloning

We train a flow-matching behavior-cloning policy on the state-based variant of the PushT benchmark introduced by Chi et al. (2023), where a 2-D end-effector pushes a T-shaped block toward a fixed target pose. The observation is the 5-dimensional state $[\mathrm{agent}_x, \mathrm{agent}_y, \mathrm{block}_x, \mathrm{block}_y, \mathrm{block}_\theta]$ and the action is the 2-dimensional target end-effector position. Following the standard receding-horizon BC setup, each example predicts an 8-step action chunk from 2 steps of observation history. Both observations and actions are MinMax-normalized over the training set. We use an $80 - 20$ train-val split. The condition $c \in \mathbb{R}^{10}$ is the flattened observation history, the target $x \in \mathbb{R}^{16}$ is the flattened action chunk and a 128-dimensional sinusoidal time embedding is used. The x_0 distribution is a standard gaussian.

The velocity field is an 8-block residual MLP with embedding width 512, feed-forward expansion factor 4 (i.e., a 2,048-dim FFN), GELU activations, LayerNorm, and dropout 0.1. Time is encoded with a 128-dim sinusoidal positional embedding with frequencies linear in $[0, 100]$ followed by an MLP. No image encoder is used (state-based variant). We train using Adam with learning rate 10^{-4}, weight decay 10^{-5}, gradient clipping at norm 10 and batch size 1024.

B.4 Additional Experiments

Policy Evaluation Section 6 reports policy-evaluation results for `Antmaze-Large` in the main paper (Figure 5). The same analysis on the remaining two OGBench environments, `Cube-Single`

and `Humanoidmaze-medium`, is shown in Figures 11 and 12 below. Both figures use the layout described in Section 6: the top row shows training loss and flow coverage at three nominal levels, and the bottom row shows the variance-vs-error scatter along with mean interval widths for the flow, split conformal, and CQR. The qualitative findings discussed in Section 6, all hold consistently across these two additional environments.

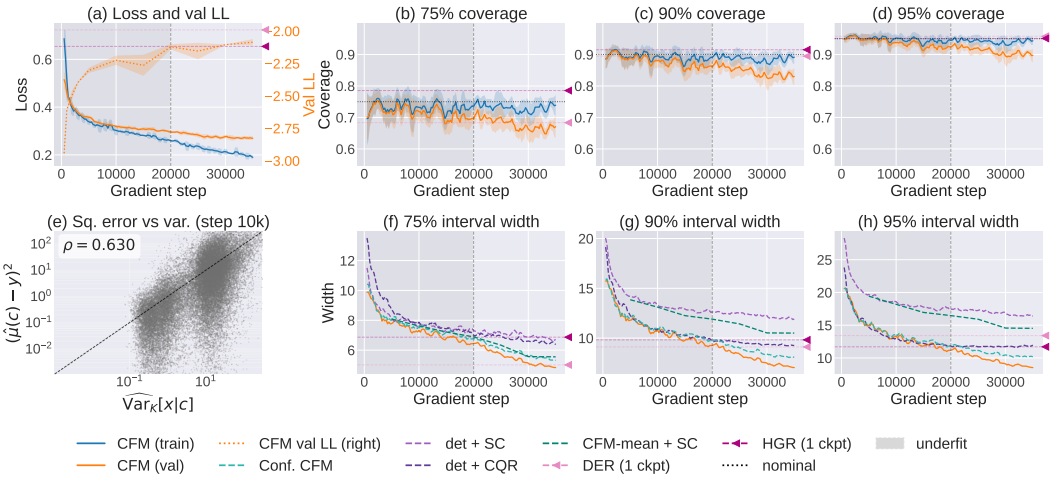


Figure 11: Coverage on `Cube-Single`. Layout matches Figure 5.

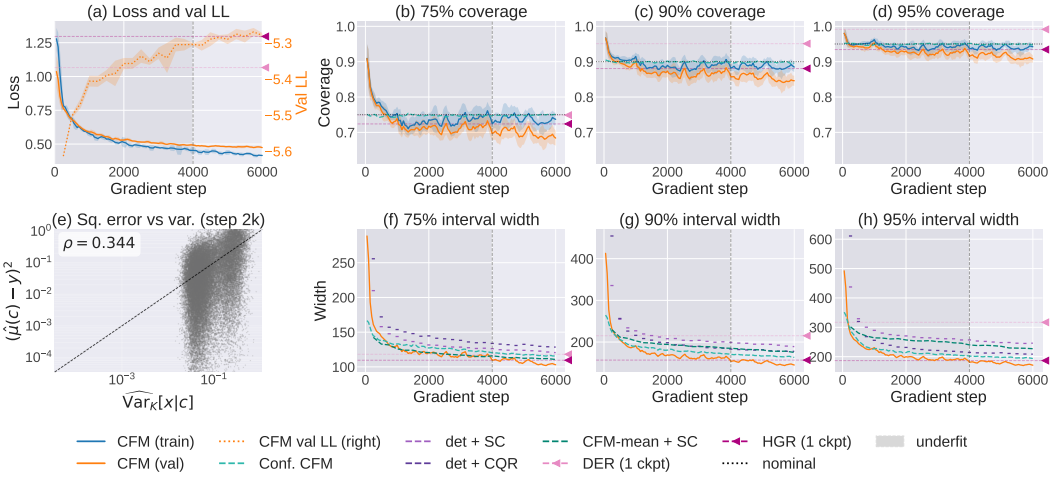


Figure 12: Coverage on `Humanoidmaze-medium`. Layout matches Figure 5.

Cell perturbation Section 6 reports coverage results for Norman in the main paper (Figure 6). The same analysis on the remaining PBMC cytokines and scPerturb benchmark SciPlex3 (drug-dose perturbations), is shown in figures 13 and 14 respectively. The figures use the same layout as Figure 6: training and validation flow-matching loss alongside KNN-based coverage at the three nominal levels.

The qualitative findings discussed in Section 6 hold here as well: training coverage remains at or above nominal throughout training, and validation coverage is close to nominal during the underfitting regime. The flow's predictive sets are approximately calibrated on SciPlex3 and PBMC, consistent with the pattern observed on Norman.

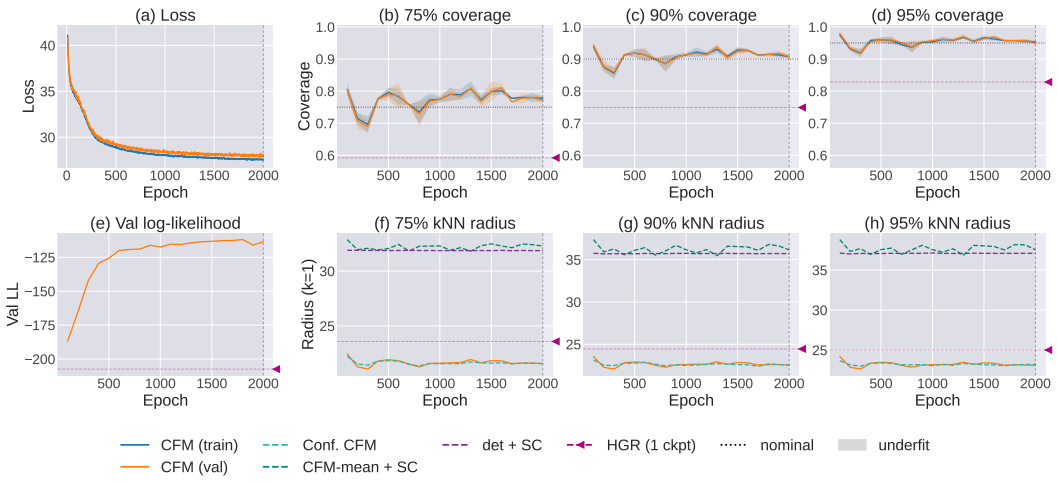


Figure 13: Coverage on PBMC. Layout matches Figure 6.

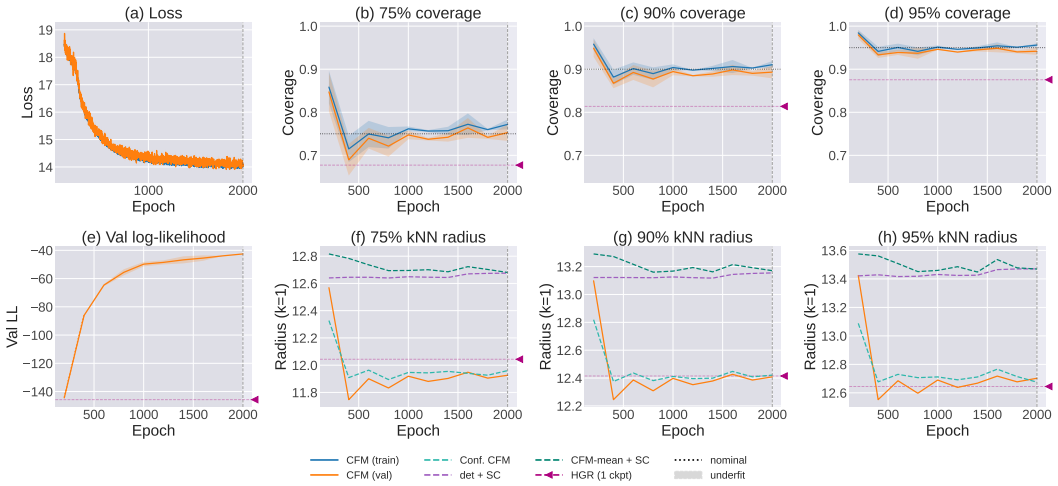
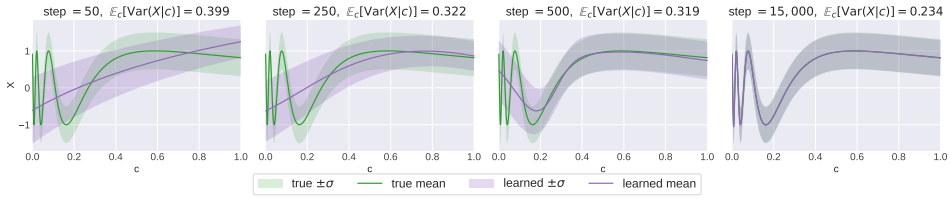


Figure 14: Coverage on SciPlex3 (drug-dose perturbations). Layout matches Figure 6.

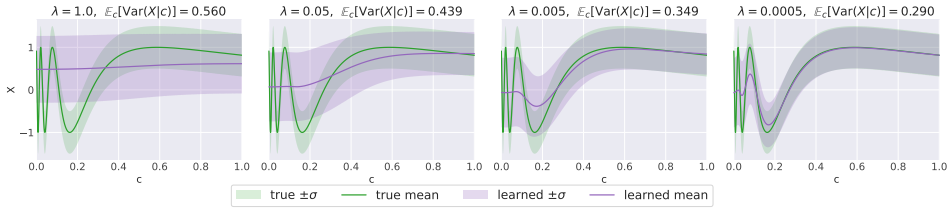
C Variance Inflation: Additional Synthetic Experiments

C.1 Variance Inflation trends on additional functions

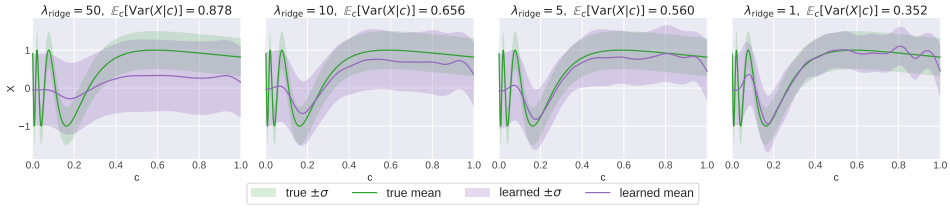
In this section, we present evidence of variance inflation trends similar to Figure 2 across several more synthetic datasets. Similar to Figure 2, for all the experiments below, the learned conditional variance increases as the learned mean becomes smoother. The average learned conditional variance decreases monotonically as smoothing decreases across all three estimators.



(a) NN vector field parameterization: [Left to Right] Distributions learned with increasing training time

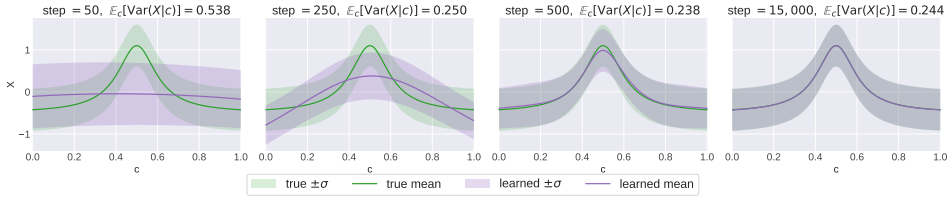


(b) NN vector field parameterization with explicit regularization penalty $\lambda(\partial_c v)^2$

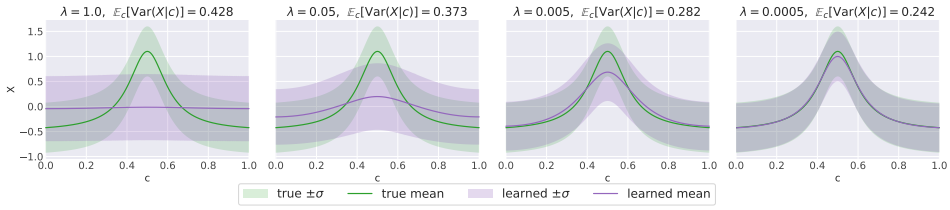


(c) Vector field learning with kernel ridge regression with penalty λ

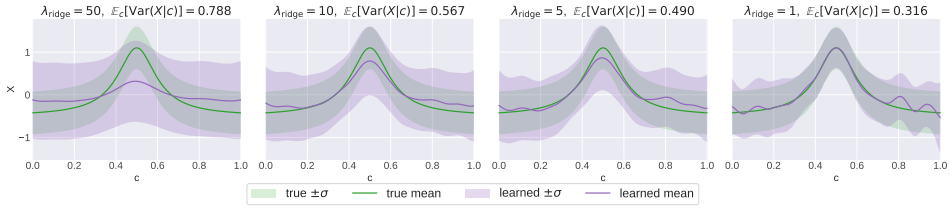
Figure 15



(a) NN vector field parameterization: [Left to Right] Distributions learned with increasing training time

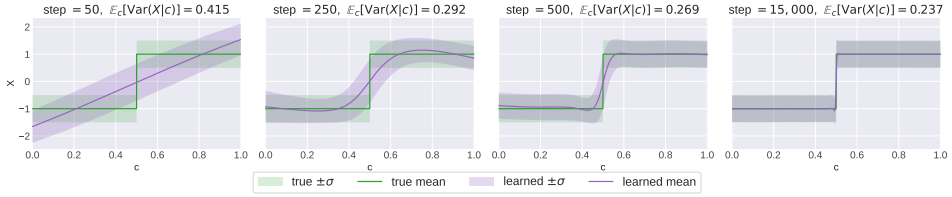


(b) NN vector field parameterization with explicit regularization penalty $\lambda(\partial_c v)^2$

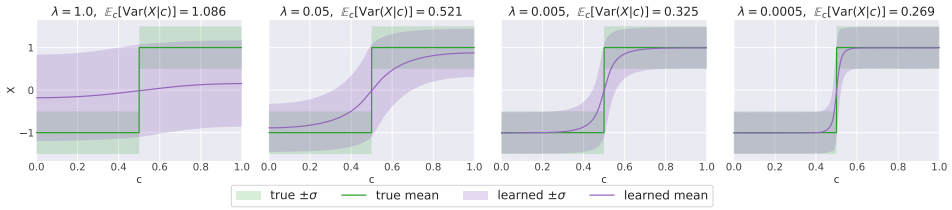


(c) Vector field learning with kernel ridge regression with penalty λ

Figure 16



(a) NN vector field parameterization: [Left to Right] Distributions learned with increasing training time



(b) NN vector field parameterization with explicit regularization penalty $\lambda(\partial_c v)^2$



(c) Vector field learning with kernel ridge regression with penalty λ

Figure 17

C.2 Excess Variance and squared bias correlation

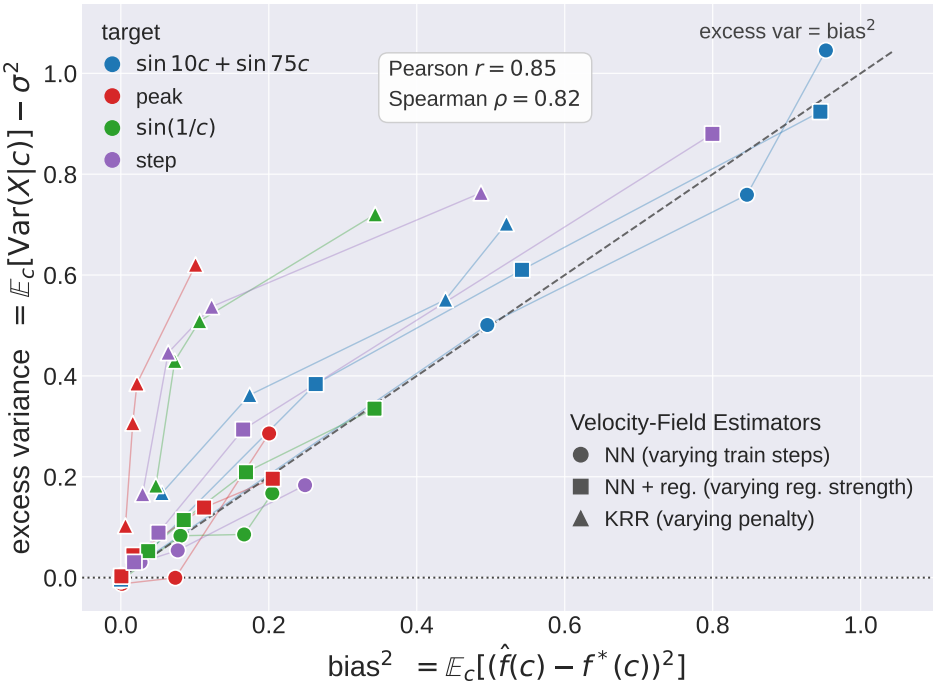


Figure 18: Excess variance learned by the trained CFM for multiple velocity-field estimators, across different distributions also plotted in: i) Section 4 Fig. 2 (blue), ii) Fig. 16 ('Peak', red), iii) Fig. 15 (green), iv) Fig. 17 ('Step', purple)