Chance-constrained Flow Matching for High-Fidelity Constraint-aware Generation

Jinhao Liang
University of Virginia
njs4nu@virginia.edu

Yixuan Sun
Argonne National Laboratory
yixuan.sun@anl.gov

Anirban Samaddar
Argonne National Laboratory
asamaddar@anl.gov

Sandeep Madireddy
Argonne National Laboratory
smadireddy@anl.gov

Ferdinando Fioretto
University of Virginia
fioretto@virginia.edu

Abstract

Generative models excel at synthesizing high-fidelity samples from complex data distributions, but they often violate hard constraints arising from physical laws or task specifications. A common remedy is to project intermediate samples onto the feasible set at each sampling step. However, repeated projection can disrupt the learned sampling dynamics because this feasible set is defined for "clean" samples rather than noisy intermediate states. Thus, recent studies first estimate "clean" samples and then project them to the feasible set, but this increases algorithmic complexity and accumulates errors across steps. To address these issues, this paper proposes *Chance-constrained Flow Matching* (CCFM), a training-free method that formulates constraint enforcement during sampling as stochastic optimization while retaining the hard-constraint feasibility guarantee of projection-based methods. By avoiding direct projection of noisy intermediate states onto the clean-sample feasible set, CCFM mitigates the distributional distortion. Unlike approaches that first estimate and project "clean" samples, CCFM avoids complex multi-stage procedures. Experiments show that CCFM outperforms current state-of-the-art constrained generative models in modeling complex physical systems governed by partial differential equations, molecular docking problems, and physics-informed motion generation, delivering superior feasibility and fidelity.

1 Introduction

Diffusion and flow-matching generative models have achieved remarkable performance across multiple domains, including image generation [17, 29, 36, 35, 34], physical system modeling [7, 48], and other scientific applications [51, 38, 9]. Although these models are highly effective at producing content that follows complex distributions, a key challenge is ensuring that generated samples strictly satisfy specific constraints. For example, models of physical systems should satisfy partial differential equations, and generated protein-ligand complexes must satisfy steric and chemical constraints.

To address these challenges, a growing literature has started exploring constrained generative models. However, these approaches present several notable limitations: *(i)* First, gradient-based guidance methods are widely used in scientific applications, yet they cannot guarantee that generated samples satisfy the imposed constraints [18, 46, 45]; *(ii)* Second, repeated projection methods can enforce strict feasibility, but they often degrade sample quality, particularly on complex tasks [8, 44]; *(iii)* Finally, a more recent line of work attempts to mitigate these issues via an *extrapolation–correction–interpolation* (ECI) procedure, which estimates "clean" samples, projects them onto the feasible set, and interpo-

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

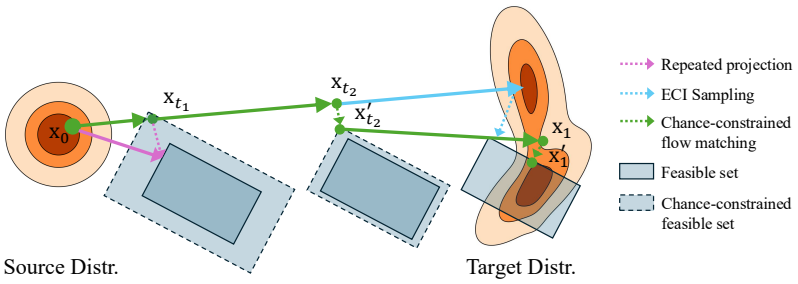


Figure 1: Overview of chance-constrained flow matching (CCFM). Solid arrows follow the sampling path; dashed arrows indicate projection steps. In CCFM, early states remain unchanged (x_{t_1}), intermediate states are minimally corrected ($x_{t_2} \to x'_{t_2}$), and the final state is guaranteed feasible (x'_1) in a high-probability region. Repeated projection distorts the sampling path. ECI sampling relies on the quality of one-step prediction and may project outside high-probability regions.

lates the corrected samples back to the noisy sampling trajectory [7, 48]. However, ECI methods complicate constraint enforcement significantly and rely heavily on accurate clean-sample estimates.

To address these challenges, this paper introduces *Chance-constrained Flow Matching* (CCFM), a training-free framework for constrained generation that guarantees feasibility while maintaining high-fidelity sample generation. Specifically, feasibility means satisfying the prescribed hard constraints, such as conservation laws in PDEs. Fidelity means staying close to the target distribution, such as realistic bound structures in molecular docking. CCFM starts by revisiting repeated projection, which applies the clean-sample feasible set to highly noisy intermediate states and thus causes overly restrictive corrections early in sampling that can degrade sample fidelity, as illustrated by the pink arrows in Figure 1. This motivates us to seek a time-dependent relaxed feasible set defined directly on intermediate states. One key observation is that, under the standard optimal transport path, each intermediate state is affine in the initial state and the clean state [31, 30]. This affine relation also serves as the premise of ECI-style methods [7]. This relation suggests pulling back the clean feasible set to the intermediate state using the initial state. However, once the original sampling trajectory is modified by a projection step, the subsequent trajectory is no longer the deterministic characteristic associated with the originally sampled x_0. Therefore, we consider a random variable drawn from the known source distribution (e.g., Gaussian) and formulate constraint enforcement as a chance-constrained program, an optimization framework widely used in operations research to enforce constraints under uncertainty with probabilistic guarantees [21]. This can yield a relaxed feasible set on noisy states that enforces constraints with a prescribed probability. Consequently, CCFM enables direct intermediate-state projection, as shown by the green arrows in Figure 1, while avoiding both the aggressive early corrections of repeated projection and complex procedures of ECI.

Contribution. This work makes the following key contributions: **(1)** It introduces *Chance-constrained Flow Matching* (CCFM), a novel framework that casts the projection step from a stochastic optimization perspective, enabling concise and effective enforcement of hard constraints while preserving the high-fidelity sample generation. **(2)** It establishes a theoretical framework for enforcing hard constraints in the sampling process of generative models from the perspective of chance-constrained programming. The analysis demonstrates that CCFM guarantees feasibility while preserving sample quality. **(3)** It demonstrates the effectiveness of CCFM across three scientific domains: molecular docking with structural constraints, physical systems modeling governed by PDEs, and physics-informed motion generation with ODEs. In all domains, CCFM produces samples that satisfy physical principles, achieving new state-of-the-art performance.

2 Related Work

Constrained generative modeling modifies either training or sampling to incorporate constraints into generated samples. We group existing methods into three classes. **(1)** *Penalty-based training* adds constraint-violation penalties to the training objective [39, 25, 1]. This approach is simple, but soft penalties rarely give hard feasibility guarantees and can be brittle in out-of-distribution settings. **(2)** *Gradient-based guidance* uses a gradient-based penalty term during sampling to steer samples toward

the feasible region [18, 20, 45]. Similar guidance is also used in Bayesian inverse problems with generative priors, where likelihood or data-consistency terms guide posterior sampling [46, 2, 22]. However, it does not guarantee feasibility and often struggles with nonconvex constraints [27]. **(3)** *Projection-based methods* enforce feasibility by projecting samples onto the feasible set [8, 9]. Repeated projection can enforce hard constraints but usually applies the clean-sample feasible set at every sampling step. However, projecting highly noisy samples at early sampling steps onto the clean-sample feasible manifold can distort the data manifold by altering the learned sampling dynamics [10, 44]. To reduce this distortion, Cheng et al. [7] and Utkarsh et al. [48] develop ECI-style samplers that apply projection only to clean samples via clean-state prediction. However, these samplers remain sensitive to errors in clean-state prediction and interpolation in complex settings. Beyond these approaches, SGEquiDiff incorporates space-group constraints into a crystal generative model through equivariant diffusion [5]. However, this approach requires a specialized model, whereas CCFM enforces constraints during sampling from a pretrained model.

In contrast, *CCFM* defines time-dependent relaxed feasible sets directly at noisy intermediate states through a chance-constrained formulation. This perspective enables direct projection during sampling without requiring clean-sample prediction. As a result, CCFM reduces projection-induced distortion of the learned sampling dynamics, simplifies hard-constraint enforcement, and preserves feasibility guarantees while maintaining high-fidelity generation.

3 Preliminaries

Before introducing CCFM, we review the concepts of generative flow matching and chance-constrained programming.

Flow matching. Flow matching (FM) is a generative framework that learns a velocity field $u_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$, which defines a mapping (*flow*) $\psi_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ that transforms samples x_0 from a source distribution p into samples $x_1 = \psi_1(x_0)$ on a target distribution q by solving the ordinary differential equation (ODE) [30]: $\frac{d}{dt}\psi_t(x) = u_t(\psi_t(x))$, with condition $\psi_0(x) = x$.

Here, the variable $t \in [0,1]$ represents the time in FM, with $t = 0$ corresponding to the initial state (noise) and $t = 1$ to the final state (clean sample). To learn the velocity field, a probability path $\{p_t\}_{t\in[0,1]}$ is defined to connect $p_0 = p$ and $p_1 = q$. This reduces distribution matching to a supervised learning problem, as the path provides intermediate states corresponding to target velocities for training u_t. The optimal transport (OT) displacement path is widely adopted as it ensures smooth and cost-minimizing interpolations between distributions that yield stable intermediate states for training [31, 23]. The OT displacement interpolation defines a marginal probability path between p and q [37, 50, 49]. Let $X_0 \sim p$ and $X_1 \sim q$ denote source and target random variables paired along the OT path. For $t \in [0,1]$, define

$$X_t = (1-t)X_0 + tX_1. \tag{1}$$

The induced law of X_t is the marginal distribution p_t. The corresponding flow-matching loss is $\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{(X_0,X_1),\, t\sim\mathrm{Unif}[0,1]} \left\| u_\theta(X_t,t) - (X_1 - X_0) \right\|^2$, where u_θ is the learnable velocity field parameterized by θ, and $\big(X_1 - X_0\big)$ denotes the target velocity. After training, the initial state is drawn by $x_0 \sim p$ and propagated through the flow ODE $\frac{d}{dt}x_t = u_\theta(x_t,t)$, which can be solved with the Euler method [19]:

$$x_{t+\Delta t} = x_t + \Delta t\, u_\theta(x_t,t). \tag{2}$$

For constrained generation, let g denote constraints defined on clean samples. The goal is to generate a final sample x_1 that satisfies $g(x_1) \le 0$ while preserving fidelity to the target distribution q.

Chance-constrained Programming. *Chance-constrained programming* (CCP) [21] is widely used to handle constraints that include terms whose exact values are unavailable but whose distributions are known. In CCFM, this setting arises during sampling. Once a correction modifies an intermediate state, the subsequent sampling trajectory is no longer determined by the initially sampled x_0. To account for this change, CCFM introduces an auxiliary source-prior variable $Z \sim p_0$ to construct a source-prior-induced relaxed feasible set at time t.

Consider an optimization problem of minimizing $f(x)$ subject to $g(x) \le 0$, where x denotes the decision variable. When the constraints depend on a random input ξ, i.e., $g(x,\xi) \le 0$, enforcing

feasibility for every realization of ξ is often too conservative or intractable. CCP instead imposes a probabilistic feasibility requirement [40, 26]:

$$\min_x f(x) \quad \text{s.t.} \quad \mathbb{P}\big(g(x,\xi) \le 0\big) \ge \alpha, \tag{3}$$

where $\alpha \in (0,1)$ is the required probability of constraint satisfaction under the distribution of ξ. Thus, any feasible solution of Problem 3 satisfies the constraint with probability at least α, rather than for every possible realization of the uncertainty. In practice, CCP has been well studied, and exact reformulations or tractable approximations often render even complex chance constraints computationally tractable [15, 14, 6, 40].

4 Chance-constrained Flow Matching

This section proposes *Chance-constrained Flow Matching* (CCFM), a novel integration of flow matching with chance-constrained programming to generate high-fidelity samples with feasibility guarantees. We first recast the classical projection method. Then, we introduce chance constraints for flow-matching and derive a tractable reformulation. Finally, we provide a theoretical analysis.

Repeated Projection. Flow matching evolves an initial noise sample toward a final clean sample via the learned vector field using an explicit Euler update (see Equation 2). To enforce feasibility, prior work often applies *repeated projections* [8, 43]. Let $\mathcal{C}_1 = \{x : g(x) \le 0\}$ denote the feasible set for clean samples. At each sampling step, these methods project the intermediate state onto this set, $\mathcal{P}_{\mathrm{rp}}(x_t) \coloneqq \arg\min_{x' \in \mathcal{C}_1} \|x' - x_t\|_2^2$, where x'_t is the nearest point in the feasible set $\mathcal{C}_1$. With this operator, the projected sampling rule in Equation 2 becomes

$$x_{t+\Delta t} = \mathcal{P}_{\mathrm{rp}}(x_t + \Delta t\, u_\theta(x_t, t))\,, \tag{4}$$

that is, each tentative Euler step is followed by a projection onto $\mathcal{C}_1$. When $\mathcal{C}_1$ is convex, this projected-Euler scheme preserves feasibility at every step [9].

4.1 Chance-constrained Flow Matching

Repeated projection applies the clean feasible set $\mathcal{C}_1$ to noisy intermediate states directly. This creates a mismatch: $\mathcal{C}_1$ constrains clean samples, while x_t still contains initial noise. In early sampling steps, projecting such noisy states onto $\mathcal{C}_1$ imposes overly tight constraints and can distort the induced distribution [24, 42]. To address this issue, *CCFM* reformulates Problem 4 via a CCP lens, allowing us to adaptively adjust the tightness of the constraints over sampling time t.

Algorithm 1: Chance-constrained Flow Matching

Input: Learned vector field u_θ, constraints g, Euler steps N.
Sample initial noise $x_0 \sim p$;
for $t \leftarrow 0, 1/N, \ldots, (N-1)/N$ **do**
  $s \leftarrow t + 1/N$;
  $y \leftarrow x_t + u_\theta(x_t, t)/N$;
  Form the chance constraints at time s with $\alpha = \phi_n(s)$;
  Reformulate using Proposition 4.2;
  Project y onto the resulting feasible set to obtain x_s;
return x_1;

Under the OT interpolation identity in Equation 1, for the fixed source realization x_0 and intermediate realization x_t, the clean terminal state x_1 can be expressed in closed form by rearranging Equation 1 as $x_1 = t^{-1}x_t - (1-t)t^{-1}x_0$. This gives the pathwise pullback of the clean feasible set to time t:

$$\mathcal{C}_t(x_0) \coloneqq \big\{x_t \in \mathbb{R}^d : g\big(t^{-1}x_t - (1-t)t^{-1}x_0\big) \le 0\big\}\,. \tag{5}$$

Thus, if x_t lies on the unmodified sampling trajectory, then $x_t \in \mathcal{C}_t(x_0)$ if and only if its corresponding clean state is feasible. However, once a projection modifies an intermediate state, the corrected state no longer lies on the original sampling trajectories associated with the initially sampled x_0. This motivates us to replace the fixed-source pullback $\mathcal{C}_t(x_0)$ with a source-prior-induced surrogate. Let $Z \sim p_0$ be an auxiliary source variable drawn from the known source distribution. For a satisfaction level $\alpha \in (0,1]$, define the chance-constrained feasible set:

$$\widehat{\mathcal{C}}_t(\alpha) \coloneqq \big\{x_t \in \mathbb{R}^d : \mathbb{P}_{Z\sim p_0}\big[g\big(t^{-1}x_t - (1-t)t^{-1}Z\big) \le 0\big] \ge \alpha\big\}\,. \tag{6}$$

The probability in Equation 6 is *not* posterior uncertainty over the historical x_0 that produced x_t. Rather, Z is an auxiliary source-prior variable used to define a time-dependent feasible set on

intermediate states. The CCFM projection is then

$$\mathcal{P}_{\mathrm{cc}}^{t}(x) \coloneqq \underset{y\in\mathbb{R}^d}{\arg\min}\ \|y-x\|_2^2 \quad \text{s.t. } \mathbb{P}_{Z\sim p_0}\left[g\big(t^{-1}y-(1-t)t^{-1}Z\big)\le 0\right]\ge \alpha. \tag{7}$$

In practice, we set $\alpha = \phi_n(t) = \frac{1}{2}t^n$ with $n > 0$. This monotone continuation schedule controls how quickly CCFM moves from loose constraints early in sampling to tighter constraints near the terminal time. Smaller n yields larger α at intermediate times and tightens the constraints earlier, whereas larger n delays tightening and keeps the sampler closer to the unconstrained flow for longer. Then, the update rule for the CCFM sampling process is given by

$$x_{t+\Delta t} = \mathcal{P}_{\mathrm{cc}}^{t+\Delta t}\left(x_t + \Delta t\, u_\theta(x_t, t)\right). \tag{8}$$

The detailed procedure of CCFM is summarized in Algorithm 1.

Compared with repeated projection [8], $\widehat{\mathcal{C}}_t(\alpha)$ relaxes the feasible region early in sampling and tightens it as t approaches 1. At $t = 0$, the sample is pure source noise and no projection is applied; in early sampling, the chance-constrained set is relaxed by the scheduler. As $t \to 1$, the source-prior uncertainty vanishes, and the chance-constrained feasible set reduces to the deterministic clean feasible set $\widehat{\mathcal{C}}_1(\alpha) = \mathcal{C}_1,\ \forall\alpha > 0$. Compared with ECI-style samplers [7, 48], CCFM also avoids the extrapolate–project–interpolate pipeline: it does not require clean-sample prediction, correction of that prediction, and interpolation back to the noisy state, thereby reducing the risk of error accumulation across sampling steps.

Remark 4.1 (Scope of the OT-path assumption). CCFM is derived under the OT displacement path in Equation 1, which is a standard geometric assumption in flow matching and has been adopted in recent constrained flow-matching samplers [7, 48]. In practice, learned velocities and numerical discretization may introduce slight curvature around the ideal straight-line characteristics. Nevertheless, recent theoretical results show that Lipschitz neural networks can approximate the ground-truth velocity field within $(\sqrt{d}+1)\epsilon$ for any $\epsilon > 0$, where d is the data dimension [52]. Discretization error also decays as the step size decreases [13]. The empirical success of straight-line characteristics in prior work [31, 32] and in our experiments further supports this assumption. Appendix C.1 evaluates 1,000 unconstrained Reaction–Diffusion trajectories. The mean normalized deviation from straight interpolation between the initial noise and clean sample is less than 1%.

4.2 Tractable Reformulation for CCFM

The projection in Equation 7 involves a probabilistic constraint. Although general chance-constrained programs are difficult, the source distribution in flow matching is typically Gaussian, which yields simple deterministic reformulations.

Proposition 4.2 (Gaussian affine chance constraint)**.** *Fix $t \in (0, 1]$ and $\alpha \in (0, 1)$. Let $a, x_t \in \mathbb{R}^d$, $b \in \mathbb{R}$, and $\xi_t = (1-t)t^{-1}Z$ with $Z \sim \mathcal{N}(0, I_d)$. Define $\sigma_t = (1-t)/t$, and let $h : \mathbb{R}^d \to \mathbb{R}$ be any deterministic function independent of ξ_t. Then*

$$\mathbb{P}_{\xi_t}\big(h(x_t) + a^\top\xi_t \le b\big) \ge \alpha \quad \Longleftrightarrow \quad h(x_t) \le b - \sigma_t\|a\|_2\Phi^{-1}(\alpha),$$

where Φ is the standard normal CDF.

Proposition 4.2 does not require h to be convex in x_t: for any deterministic h independent of ξ_t, affine-in-noise chance constraints admit the exact deterministic reformulation. CCFM therefore can handle these chance constraints with the same projection machinery used for deterministic constraints. Since $z_\alpha = \Phi^{-1}(\alpha)$ is monotone in α and the standard deviation $\sigma_t = \frac{1-t}{t}$ decreases as $t \to 1$, the constraint bound tightens progressively with the sampling time t. Appendix A provides the proof.

For nonlinear dependence on the uncertainty, we use a local first-order stochastic surrogate, a standard approximation in nonlinear and stochastic optimization [6, 40, 41]. Given a reference value $\bar{\xi}_t$ of the random variable ξ_t, e.g., $\bar{\xi}_t = \mathbb{E}[\xi_t]$, we linearize

$$g(y, \xi_t) \approx g(y, \bar{\xi}_t) + \nabla_\xi g(y, \bar{\xi}_t)^\top(\xi_t - \bar{\xi}_t).$$

Since this approximation is affine in the Gaussian random variable ξ_t, Proposition 4.2 applies and yields a deterministic reformulation of the corresponding chance constraint.

Following the existing constrained flow matching [48], we solve the resulting deterministic projection problem with an active-set Gauss–Newton update. Let $\mathcal{A}$ be the active constraints, $r_{\mathcal{A}}$ denotes the residual vector, and $J_{\mathcal{A}} = \partial r_{\mathcal{A}}/\partial y$. We update

$$y \leftarrow y - J_{\mathcal{A}}^{\top}\left(J_{\mathcal{A}}J_{\mathcal{A}}^{\top} + \lambda I\right)^{-1} r_{\mathcal{A}},$$

where $\lambda > 0$ is a small regularizer.

4.3 Theoretical Analysis for CCFM

Having established a tractable chance-constrained projection for any time $t \in (0, 1]$, the next result shows that this procedure guarantees the feasibility of generated samples.

Corollary 4.3 (Feasibility Guarantee)**.** *For any satisfaction level $\alpha \in (0, 1]$, the chance-constrained projection in Equation 7 satisfies*

$$\mathcal{P}^1_{\mathrm{cc}}(x_1) = \arg\min_{y \in \mathcal{C}_1} \|y - x_1\|_2^2, \qquad \mathcal{C}_1 = \{y \in \mathbb{R}^d : g(y) \le 0\}.$$

Proposition 4.2 and Corollary 4.3 imply that the operator is probabilistic for $t < 1$ and becomes deterministic at $t = 1$, ensuring feasibility. However, it is worth noting that this exact feasibility guarantee applies to the constraint set $\mathcal{C}_1$. In practice, for computational efficiency, some highly complex constraints, such as the molecular docking constraints in our experiments, are implemented using standard approximations of the original constraint set. Despite this, the results show that our method remains robust in realistic constrained-generation settings.

Beyond feasibility, we quantify how much CCFM perturbs the unconstrained learned dynamics.

Theorem 4.4 (Correction-controlled perturbation)**.** *Let $h = 1/N$, $t_k = kh$, and $F_k(x) = x + hu_\theta(x, t_k)$. Assume $u_\theta(\cdot, t_k)$ is L-Lipschitz uniformly in k, and set $\gamma = 1 + hL$. Consider unconstrained sampler $Y_{k+1} = F_k(Y_k)$ and constrained sampler $X_{k+1} = P_k(F_k(X_k)) = F_k(X_k) + e_k$ with $X_0 = Y_0$, where P_k is a projection operator. Then*

$$\|X_N - Y_N\|_2 \le \sum_{k=0}^{N-1} \gamma^{N-1-k}\|e_k\|_2. \tag{9}$$

If $X_0 = Y_0 \sim p_0$, the final distributions have finite second moments, and $\mathbb{E}\|e_k\|_2^2 < \infty$, then the L^2-Wasserstein distance

$$W_2(\mu_N, \mu_N^{\mathrm{free}}) \le \textstyle\sum_{k=0}^{N-1} \gamma^{N-1-k}\left(\mathbb{E}\|e_k\|_2^2\right)^{1/2}, \tag{10}$$

where μ_N and μ_N^{free} are the distributions of X_N and Y_N, respectively.

Theorem 4.4 shows that distributional shift from the learned unconstrained dynamics is controlled by the correction magnitudes. Since the weights γ^{N-1-k} are non-increasing in k, early corrections are propagated through more subsequent Euler steps and can induce larger trajectory distortion. This supports the design of CCFM, whose relaxed constraint sets avoid overly restrictive projections at noisy intermediate states and tighten only as $t \to 1$. Furthermore, for distributions with finite second moments, exact Euclidean projection onto a nonempty closed convex set cannot increase the W_2 distance to any target distribution supported on that set [3].

5 Experiments

This section first considers a two-dimensional example with a known constrained target distribution, allowing direct evaluation of distributional fidelity through W_2 distance. We then evaluate CCFM on three scientific tasks: **Molecular Docking**, **PDE Solution Generation**, and **Physics-informed Motion Generation**. Further details on the experimental settings are provided in Appendix B. Additional results are presented in Appendix C.

5.1 Experimental Settings

Two-Dimensional Example. We consider the initial noise $X_0 \sim \mathcal{N}(0, I_2)$ and a Gaussian mixture target distribution $q = 0.3\mathcal{N}((0, -1), 0.7^2 I_2) + 0.7\mathcal{N}((2, 1), 0.7^2 I_2)$. For a clean sample $z =$

$(z_1, z_2) \in \mathbb{R}^2$, the feasible set is $\mathcal{C} = \{z : z_1 = -1\} \cup \{z : z_1 = 1\}$. Specifically, these two branches have unequal probability mass under the ground-truth constrained distribution q_C. Recovering this mass allocation tests whether a sampler preserves distributional structure while satisfying constraints.

Comparisons are made against *Repeated Projection (RP)*, *ECI*, and *Post-only Projection*, which projects only the final sample. Evaluation uses W_2 distance to q_C and branch-mass error, which measures discrepancies in the probabilities assigned to the two feasible branches.

Molecular Docking. The task is to predict ligand poses within protein pockets while respecting geometric and physical constraints. Experiments are performed on the PDBBind docking benchmark [33], which provides a standard testbed for evaluating protein–ligand docking methods. All methods are evaluated on the same 363 test complexes. Reported success rates are computed over these complexes. Comparisons are made against a *repeated projection (RP)* baseline and *FlexDock* [11], the current state-of-the-art docking framework, which enforces structural constraints by embedding penalty terms during training and further ensuring validity at inference through rejection sampling.

The evaluation focuses on: **(1)** *Fidelity*, which is quantified by ligand RMSD (L-RMSD) (fractions below 1Åand 2Å) and all-atom RMSD (A-RMSD) (fractions below 2Å). **(2)** *Feasibility*, which is assessed by PoseBusters validity (PB Valid) [4] and its conjunction with RMSD thresholds, measuring the physical plausibility of predicted complexes.

PDE Solution Generation. Next, following [48], we consider a generative task on two representative PDE systems: the 1-D *reaction–diffusion equation*, subject to linear initial conditions (IC) and nonlinear global mass conservation laws (CL), and the 2-D *Navier–Stokes equation*, subject to linear initial conditions (IC) and linear conservation laws (CL). The reaction–diffusion problem is discretized on a (128×100) space–time grid, while the Navier–Stokes problem is posed on a $64 \times 64 \times 50$ grid. For each PDE, all methods are evaluated on the same 1,000 test trajectories.

Comparisons are conducted against several representative baselines: (a) *Physics-Constrained Flow Matching (PCFM)*[48], which is the current state of the art; it extends ECI sampling [7] to nonlinear constraints and reduces to ECI in the linear setting; (b) *DiffusionPDE* [18], which enforces physical priors through penalty terms during iterative sampling; (c) *Functional Flow Matching (FFM)* [23], a pretrained unconstrained FM; and (d) *Repeated Projection (RP)* approach, which naively projects the state onto the feasible set at each sampling step.

Performance is evaluated along two complementary dimensions: **(1)** *Feasibility* is quantified by the constraint violation (CV), which measures deviations from the imposed boundary, initial, and conservation constraints. **(2)** *Fidelity* is evaluated using the pointwise mean squared error of the mean (MMSE) and the pointwise mean squared error of the standard deviation (SMSE), capturing distributional alignment in terms of first- and second-order statistics.

Physics-informed motion. Following the setting introduced in [8], this task examines whether a generative model can synthesize video sequences that respect physical rules. The objective is to generate multi-frame trajectories of a moving object whose positions must satisfy discrete ordinary differential equation (ODE) updates coupling the position and velocity across time steps. Training data include only Earth-gravity motion, while test cases with Moon gravity provide an out-of-distribution evaluation. Comparisons are conducted against: (a) an *unconstrained* flow model, which follows the standard sampling process without any constraint enforcement; (b) a *repeated projection (RP)* baseline, which projects each intermediate sample onto the constraint set.

Evaluation focuses on two aspects: (1) Feasibility, measured by the violation of the ODE constraints across frames; (2) Fidelity, assessed by the Fréchet Inception Distance (FID), which reflects the visual quality of generated sequences under Moon gravity.

5.2 Performance Evaluation for the Two-Dimensional Example

Empirical W_2 measures the distance to the ground-truth constrained distribution, while branch-mass error measures the mismatch in probability assigned to each feasible branch. Table 1 shows that CCFM achieves the lowest error on both metrics. Repeated projection and ECI commit to a

Table 1: Two-dimensional example results over five runs.

Method	W_2 distance (↓)	Branch-mass Error (↓)
RP	1.026	0.231
ECI	2.695	0.152
Post-only projection	0.671	0.080
CCFM (ours)	**0.412**	**0.0248**

Table 2: PDBBind docking results. #S denotes the number of samples generated per complex, with the best pose selected for evaluation; #Stp is the number of inference steps. L-RMSD/A-RMSD report fractions of complexes below the given thresholds, PB Valid is the proportion passing PoseBusters checks, with +L and +A further requiring L-RMSD$<$ 2Å or A-RMSD$<$ 2Å, respectively.

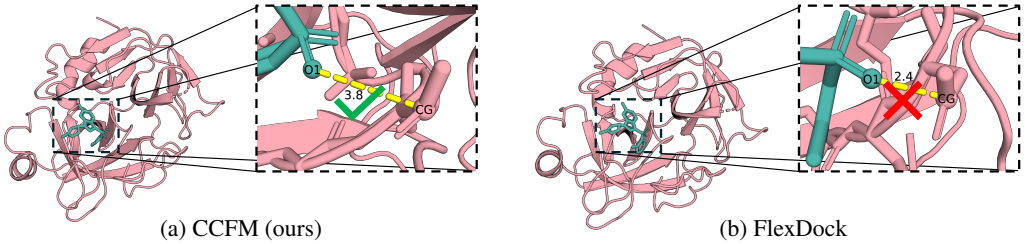
#S	#Stp	Methods	L-RMSD (%)		A-RMSD (%)	PB Valid (%)			Running
			$<$ 1Å	$<$ 2Å	$<$ 2Å	All	+L $<$ 2Å	+A $<$ 2Å	Time (s)
1	2	CCFM	**8.19**	**32.20**	**74.01**	**47.74**	**22.03**	**35.88**	0.64
		FlexDock	7.06	31.36	73.73	20.34	9.32	14.97	0.52
		RP	6.78	31.07	73.73	19.49	9.32	14.69	0.73
10	2	CCFM	**14.69**	37.01	**74.86**	**67.51**	**31.36**	**51.98**	4.34
		FlexDock	12.99	**38.14**	73.16	43.50	21.19	33.05	4.20
		RP	13.56	36.72	73.16	44.07	21.19	34.18	4.39
20	2	CCFM	12.43	**39.27**	**74.01**	**67.80**	**33.62**	**51.41**	8.51
		FlexDock	**12.71**	**39.27**	72.88	44.63	23.16	33.05	7.82
		RP	11.58	37.85	72.88	49.72	25.42	37.85	9.21
1	10	CCFM	**7.91**	**32.20**	72.32	**78.81**	**30.79**	**58.19**	0.79
		FlexDock	7.63	31.64	**72.88**	57.91	24.29	40.68	0.57
		RP	**7.91**	31.92	**72.88**	56.21	27.40	42.66	0.92
1	20	CCFM	**8.19**	**31.92**	72.32	**76.84**	**29.94**	**57.06**	0.88
		FlexDock	7.63	31.07	72.60	60.73	25.99	43.50	0.62
		RP	7.91	31.36	**72.88**	50.85	26.55	40.68	1.10
10	10	CCFM	**15.54**	37.85	72.03	**85.03**	**35.88**	**62.15**	4.57
		FlexDock	**15.54**	**38.14**	72.60	80.51	35.31	57.06	3.99
		RP	13.84	**38.14**	**72.88**	76.55	34.75	56.78	5.15

branch too early, whereas post-only projection retains biased branch weights from unconstrained sampling. By delaying branch commitment, CCFM more accurately recovers the constrained distribution. Its improvement over post-only projection shows that terminal feasibility alone does not ensure distributional fidelity.

5.3 Performance Evaluation for Molecular Docking

Table 2 reports docking results on the PDBBind docking benchmark under varying numbers of generated samples and inference steps. Across all configurations, *CCFM* consistently outperforms the baselines *FlexDock* and *RP*, *with particularly large gains when the computational budget is limited*. Remarkably, with only one sample and two inference steps, CCFM achieves comparable L-RMSD and A-RMSD values, confirming that high-fidelity pose generation is preserved. At the same time, CCFM raises PB Valid to 47.74%, representing a **2.3**$\times$ improvement over FlexDock (20.34%) and **2.4**$\times$ over RP (19.49%). In contrast to RP, CCFM avoids the performance degradation arising from the application of strict constraints to noisy intermediate states. With this setting, CCFM even surpasses FlexDock with twenty samples, yielding a **20**$\times$ improvement in sample–step efficiency (measured as samples $\times$ steps) and a **12**$\times$ runtime speedup (7.82s vs. 0.64s). On stricter metrics that jointly account for fidelity and feasibility (PB Valid+L and PB Valid+A), CCFM also achieves more than $2\times$ higher pass rates than both baselines, again matching FlexDock at 20 samples while using far fewer samples and computational time. Increasing inference steps to 10 or 20 further amplifies this advantage: at 10 steps, CCFM improves PB Valid by 20.9%, PB Valid+L by 6.5%, and PB Valid+A by 17.5%, while simultaneously yielding higher fidelity and feasibility than the baseline. Comparable trends are observed when both the sample count and inference steps are set to ten. Additional comparisons across more metrics and experimental settings are provided in Appendix C.2.

Figure 2 compares the binding structures generated by our method (CCFM) and FlexDock on the same protein–ligand pair. In molecular docking, protein–ligand atoms must stay at least the sum of their van der Waals radii minus 0.75 Å [11]. In this case, the van der Waals radii of the carbon (1.70 Å) and oxygen (1.52 Å) atoms yield a lower bound of 2.47 Å. As shown in Figure 2a, CCFM produces a feasible structure with a distance of 3.8 Å , whereas FlexDock in Figure 2b generates an infeasible structure with a distance of 2.4 Å.



(a) CCFM (ours) (b) FlexDock

Figure 2: Structures generated by CCFM (ours) and FlexDock. CCFM enforces the ligand–protein distance constraint, while FlexDock violates it.

Table 3: Comparison of model performance in solution generation tasks for 1D Reaction–Diffusion equation and 2D Navier–Stokes equation across metrics. A lower value indicates better performance. Bold indicates the best (smallest) value in each row.

Problem	Metric	CCFM	PCFM	DPDE	FFM	RP
RD	MMSE $\times 10^{-2}$ (↓)	**3.3**	3.6	5.2	5.3	8.1
	SMSE $\times 10^{-2}$ (↓)	**2.7**	2.9	3.8	3.8	8.5
	CV (IC) $\times 10^{-2}$ (↓)	**0**	**0**	8.0	8.2	**0**
	CV (CL) $\times 10^{-15}$ (↓)	**9.5**	**9.5**	3.9×10^{10}	3.9×10^{10}	10.8
NS	MMSE $\times 10^{-1}$ (↓)	**0.8**	1.7	1.4	1.5	1.0
	SMSE $\times 10^{-1}$ (↓)	**0.8**	1.6	8.8	9.0	0.9
	CV (IC) (↓)	**0**	**0**	0.12	0.13	**0**
	CV (CL) $\times 10^{-15}$ (↓)	3.9	**3.4**	1.2×10^{10}	1.2×10^{9}	4.1

5.4 Performance Evaluation for PDE

We next focus on generating PDE solutions. Table 3 summarizes the performance of the different models tested. For the Reaction–Diffusion, the unconstrained *FFM* baseline yields relatively poor accuracy (MMSE 5.3×10^{-2}), while DiffusionPDE improves marginally (to 5.2×10^{-2}) by incorporating penalty terms during sampling that steer samples towards the prescribed initial condition and enforce the conservation law. However, this approach fails to handle non-linear constraints and does not guarantee feasibility. The *RP* baseline successfully enforces the initial condition constraint, yet this comes at the cost of solution fidelity, yielding the worst accuracy among all methods. *PCFM* achieves a substantial accuracy gain (MMSE reduced to 3.6×10^{-2}) while simultaneously ensuring minimal constraint violations. Nevertheless, this procedure incurs a computational burden (see Appendix C) due to the projections at every sampling step, resulting in a runtime that is approximately $28\times$ higher than the unconstrained FFM baseline. *CCFM* further advances performance (MMSE 3.3×10^{-2}, SMSE 2.7×10^{-2}) while providing **zero constraint violations**. Moreover, *CCFM* is about 30% faster than PCFM (see Appendix C). Figure 3 further illustrates its MMSE improvement across physical time, where darker regions indicate improved temporal stability.

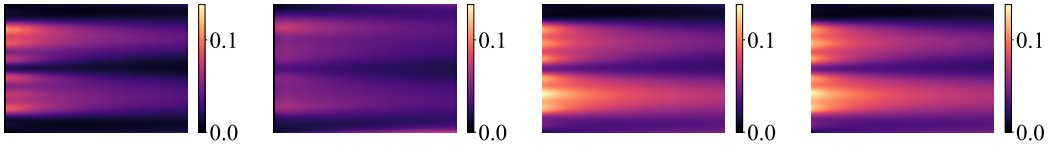
On the Navier–Stokes task, the increased dimensionality and the out-of-distribution test setting exacerbate the limitations of the baseline methods. *PCFM* yields higher MMSE than DiffusionPDE despite preserving feasibility. Its reliance on one-step predictions from intermediate states leads to inaccuracies in high dimensions, and projection–interpolation steps amplify these errors, reducing accuracy. *RP* demonstrates strong performance, outperforming both PCFM and the unconstrained baselines. However, it still lags behind *CCFM*. By making only small corrections toward the feasible region and avoiding unstable one-step extrapolations, *CCFM* achieves the best overall performance (MMSE 8×10^{-2}, SMSE 8×10^{-2}) while ensuring feasibility; Figure 4 shows that it suppresses error accumulation and aliasing under nonlinear dynamics.

5.5 Performance Evaluation for Physics-informed Motion

Table 4 shows that CCFM achieves the lowest constraint violation and improved FID compared to the RP approach. The unconstrained model attains better FID because it avoids background artifacts despite its large constraint violation. Although RP and CCFM do not directly modify the background, the Earth-trained base model tends

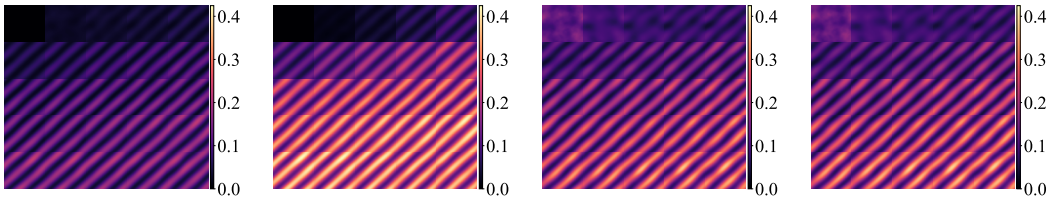
Table 4: Comparison of model performance in the physics-informed motion task.

	Unconstrained	RP	CCFM
Constraint Violation (MSE) (↓)	278.46	0.8333	**0.0100**
FID (↓)	**16.985**	27.184	26.767



(a) CCFM (ours) (b) PCFM (c) DPDE (d) FFM

Figure 3: MMSE (the darker the better) of Reaction–Diffusion solutions over physical time, with the horizontal axis representing time evolution and the vertical axis indicating errors at each point.



(a) CCFM (ours) (b) PCFM (c) DPDE (d) FFM

Figure 4: MMSE (the darker the better) of Navier–Stokes solutions over physical time, downsampled from 50 to 25 frames for clarity. Each frame corresponds to the error at a given physical time.

to generate in-distribution frames. After projection moves the object to the Moon-gravity location, the model may partially pull it back toward its in-distribution trajectory, creating slight background ghosting. This also contributes to the higher violation observed for RP, since violation is computed from detected object centers. Ghosting artifacts can interfere with center detection, increasing the measured violation.

6 Conclusion

This paper introduced *Chance-constrained Flow Matching* (CCFM), a training-free framework for constraint-aware generative modeling. Its key idea is a reformulation of the repeated projection *during flow-matching sampling* through a stochastic-optimization lens. Under exact terminal projection, CCFM guarantees terminal feasibility. Our analysis also bounds the perturbation introduced by constraint corrections relative to the unconstrained sampler. Experiments on molecular docking, PDE solution generation, and physics-informed motion demonstrate its effectiveness, with particularly strong gains in docking under limited sampling budgets.

Limitations and future work. Despite the strong potential of CCFM, several directions remain open. While Theorem 4.4 provides a finite-horizon stability bound relative to the unconstrained sampler, the convergence of the generated distribution to the exact constrained target distribution q_C remains a direction for future work. In domains such as robotics and other scientific simulation applications, constraints may not admit closed forms. This aspect is out of the scope of the work, and the challenge of embedding non-differentiable black-box simulators (e.g., collision checkers, invariants) remains an interesting direction for future work.

Acknowledgment

The material is based upon work supported by National Science Foundations (NSF) awards 2533631, 2401285, 2334448, and 2334936, and Defense Advanced Research Projects Agency (DARPA) under Contract No. #HR0011252E005. The research was also supported by the U.S. Department of Energy, Advanced Scientific Computing Research, through the SciDAC-RAPIDS2 institute, and Competitive Portfolios project on Energy Efficient Computing: A Holistic Methodology, under contract DE-AC02-06CH11357. The authors acknowledge Research Computing at the University of Virginia, the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357, Laboratory Computing Resource Center (LCRC) at the Argonne National Laboratory for providing computational resources that have contributed to the results reported within this paper. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF, DARPA, or DOE. The authors declare no competing interests.

References

[1] Giacomo Baldan, Qiang Liu, Alberto Guardone, and Nils Thuerey. Flow matching meets pdes: A unified framework for physics-constrained generation. *arXiv preprint arXiv:2506.08604*, 2025.

[2] Lorenzo Baldassari, Ali Siahkoohi, Josselin Garnier, Knut Solna, and Maarten V de Hoop. Conditional score-based diffusion models for bayesian inference in infinite dimensions. *Advances in Neural Information Processing Systems*, 36:24262–24290, 2023.

[3] Matthieu Blanke, Yongquan Qu, Sara Shamekh, and Pierre Gentine. Strictly constrained generative modeling via split augmented langevin sampling. In *International Conference on Learning Representations*, volume 2026, pages 145927–145965, 2026.

[4] Martin Buttenschoen, Garrett M Morris, and Charlotte M Deane. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 15(9):3130–3139, 2024.

[5] Rees Chang, Angela Pak, Alex Guerra, Ni Zhan, Nick Richardson, Elif Ertekin, and Ryan Adams. Space group equivariant crystal diffusion. *Advances in Neural Information Processing Systems*, 38:72772–72805, 2026.

[6] Peng Chen and Omar Ghattas. Taylor approximation for chance constrained optimization problems governed by partial differential equations with high-dimensional random parameters. *SIAM/ASA Journal on Uncertainty Quantification*, 9(4):1381–1410, 2021.

[7] Chaoran Cheng, Boran Han, Danielle C Maddix, Abdul Fatir Ansari, Andrew Stuart, Michael W Mahoney, and Yuyang Wang. Gradient-free generation for hard-constrained systems. *arXiv preprint arXiv:2412.01786*, 2024.

[8] Jacob K Christopher, Stephen Baek, and Nando Fioretto. Constrained synthesis with projected diffusion models. *Advances in Neural Information Processing Systems*, 37:89307–89333, 2024.

[9] Jacob K Christopher, Michael Cardei, Jinhao Liang, and Ferdinando Fioretto. Neuro-symbolic generative diffusion models for physically grounded, robust, and safe generation. *arXiv preprint arXiv:2506.01121*, 2025.

[10] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022.

[11] Gabriele Corso, Vignesh Ram Somnath, Noah Getz, Regina Barzilay, Tommi Jaakkola, and Andreas Krause. Composing unbalanced flows for flexible docking and relaxation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[12] Peter Eastman, Jason Swails, John D Chodera, Robert T McGibbon, Yutong Zhao, Kyle A Beauchamp, Lee-Ping Wang, Andrew C Simmonett, Matthew P Harrigan, Chaya D Stern, et al. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology*, 13(7):e1005659, 2017.

[13] Yuan Gao, Jian Huang, Yuling Jiao, and Shurong Zheng. Convergence of continuous normalizing flows for learning probability distributions. *arXiv preprint arXiv:2404.00551*, 2024.

[14] Michael A Gelbart, Jasper Snoek, and Ryan P Adams. Bayesian optimization with unknown constraints. *arXiv preprint arXiv:1403.5607*, 2014.

[15] Xinbo Geng and Le Xie. Data-driven decision making in power systems with probabilistic guarantees: Theory and applications of chance-constrained optimization. *Annual reviews in control*, 47:341–363, 2019.

[16] Jonathan E Guyer, Daniel Wheeler, and James A Warren. Fipy: Partial differential equations with python. *Computing in Science & Engineering*, 11(3):6–15, 2009.

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

[18] Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation. *Advances in Neural Information Processing Systems*, 37: 130291–130323, 2024.

[19] Arieh Iserles. *A first course in the numerical analysis of differential equations*. Cambridge university press, 2009.

[20] Christian Jacobsen, Yilin Zhuang, and Karthik Duraisamy. Cocogen: Physically-consistent and conditioned score-based generative models for forward and inverse problems, 2024. URL `https://arxiv.org/abs/2312.10527`.

[21] Peter Kall, Stein W Wallace, and Peter Kall. *Stochastic programming*, volume 5. Springer, 1994.

[22] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in neural information processing systems*, 35:23593–23606, 2022.

[23] Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Functional flow matching. *arXiv preprint arXiv:2305.17209*, 2023.

[24] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *Advances in Neural Information Processing Systems*, 37:122458–122483, 2024.

[25] Xinpeng Li, Enming Liang, and Minghua Chen. Gauge flow matching for efficient constrained generative modeling over general convex set. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025.

[26] Jinhao Liang, Wenqian Jiang, Chenbei Lu, and Chenye Wu. Joint chance-constrained unit commitment: Statistically feasible robust optimization with learning-to-optimize acceleration. *IEEE Transactions on Power Systems*, 39(5):6508–6521, 2024.

[27] Jinhao Liang, Jacob K Christopher, Sven Koenig, and Ferdinando Fioretto. Simultaneous multi-robot motion planning with projected diffusion models. *arXiv preprint arXiv:2502.03607*, 2025.

[28] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022: 500902, 2022.

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

[30] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

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

[32] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *The Twelfth International Conference on Learning Representations*, 2023.

[33] Zhihai Liu, Minyi Su, Li Han, Jie Liu, Qifan Yang, Yan Li, and Renxiao Wang. Forging the basis for developing protein–ligand interaction scoring functions. *Accounts of chemical research*, 50 (2):302–309, 2017.

[34] Chenrui Ma. Transition flow matching. *arXiv preprint arXiv:2603.15689*, 2026.

[35] Chenrui Ma, Xi Xiao, Lin Zhao, Xinru Jiang, Tianyang Wang, Yanning Shen, and Ferdinando Fioretto. Posterior flow matching. *arXiv preprint arXiv:2610.13994*, 2026.

[36] Chenrui Ma, Xi Xiao, Lin Zhao, Tianyang Wang, Ferdinando Fioretto, and Yanning Shen. Drift flow matching. *arXiv preprint arXiv:2605.17244*, 2026.

[37] Robert J McCann. A convexity principle for interacting gases. *Advances in mathematics*, 128 (1):153–179, 1997.

[38] Benjamin Kurt Miller, Ricky TQ Chen, Anuroop Sriram, and Brandon M Wood. Flowmm: Generating materials with riemannian flow matching. *arXiv preprint arXiv:2406.04713*, 2024.

[39] Saeid Naderiparizi, Xiaoxuan Liang, Berend Zwartsenberg, and Frank Wood. Constrained generative modeling with manually bridged diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19607–19615, 2025.

[40] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2007.

[41] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 2006.

[42] Kevin Rojas, Ye He, Chieh-Hsin Lai, Yuta Takida, Yuki Mitsufuji, and Molei Tao. Theory-informed improvements to classifier-free guidance for discrete diffusion models. *arXiv preprint arXiv:2507.08965*, 2025.

[43] Ralf Römer, Alexander von Rohr, and Angela P Schoellig. Diffusion predictive control with constraints. *arXiv preprint arXiv:2412.09342*, 2024.

[44] Javier E Santos, Agnese Marcato, Roman Colman, Nicholas Lubbers, and Yen Ting Lin. Discrete spatial diffusion: Intensity-preserving diffusion modeling. *arXiv preprint arXiv:2505.01917*, 2025.

[45] Yorai Shaoul, Itamar Mishani, Shivam Vats, Jiaoyang Li, and Maxim Likhachev. Multi-robot motion planning with diffusion models. *arXiv preprint arXiv:2410.03072*, 2024.

[46] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. *arXiv preprint arXiv:2111.08005*, 2021.

[47] Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Regina Barzilay, and Tommi Jaakkola. Equibind: Geometric deep learning for drug binding structure prediction. In *International conference on machine learning*, pages 20503–20521. PMLR, 2022.

[48] Utkarsh Utkarsh, Pengfei Cai, Alan Edelman, Rafael Gomez-Bombarelli, and Christopher Vincent Rackauckas. Physics-constrained flow matching: Sampling generative models with hard constraints. *arXiv preprint arXiv:2506.04171*, 2025.

[49] Cédric Villani. *Topics in optimal transportation*, volume 58. American Mathematical Soc., 2021.

[50] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2008.

[51] Jason Yim, Andrew Campbell, Emile Mathieu, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Frank Noé, et al. Improved motif-scaffolding with se (3) flow matching. *ArXiv*, pages arXiv–2401, 2024.

[52] Zhengyu Zhou and Weiwei Liu. An error analysis of flow matching for deep generative modeling. In *Forty-second International Conference on Machine Learning*, 2025.

A Missing Proofs

A.1 Proof of Proposition 4.2

Proof of Proposition 4.2. Let

$$\sigma_a := \sigma_t \|a\|_2.$$

Since $\xi_t \sim \mathcal{N}(0, \sigma_t^2 I_d)$, we have

$$a^\top \xi_t \sim \mathcal{N}(0, \sigma_a^2).$$

If $\sigma_a = 0$, then $a^\top \xi_t = 0$ almost surely. Hence,

$$\mathbb{P}_{\xi_t}\big(h(x_t) + a^\top \xi_t \le b\big) \ge \alpha$$

holds if and only if $h(x_t) \le b$, which is the stated deterministic inequality because $\sigma_t \|a\|_2 = 0$.

Assume $\sigma_a > 0$. Since $h(x_t)$ is deterministic with respect to ξ_t,

$$\mathbb{P}_{\xi_t}\big(h(x_t) + a^\top \xi_t \le b\big) = \mathbb{P}\big(a^\top \xi_t \le b - h(x_t)\big) = \Phi\left(\frac{b - h(x_t)}{\sigma_a}\right).$$

Because Φ is strictly increasing and $\alpha \in (0, 1)$,

$$\Phi\left(\frac{b - h(x_t)}{\sigma_a}\right) \ge \alpha \quad \Longleftrightarrow \quad \frac{b - h(x_t)}{\sigma_a} \ge \Phi^{-1}(\alpha).$$

Rearranging gives

$$h(x_t) \le b - \sigma_a \Phi^{-1}(\alpha) = b - \sigma_t \|a\|_2 \Phi^{-1}(\alpha),$$

which proves the equivalence. □

A.2 Proof of Corollary 4.3

Proof. From Equation 6, the random term is $\xi_t = (1 - t)t^{-1}Z$. At $t = 1$, it follows that $\xi_t = 0$. Hence, for any candidate y,

$$\mathbb{P}_{Z \sim p_0}\big(g(t^{-1}y - \xi_t) \le 0\big) = \mathbb{P}_{Z \sim p_0}(g(y) \le 0) = \begin{cases} 1, & g(y) \le 0, \\ 0, & \text{otherwise.} \end{cases}$$

Since $\alpha \in (0, 1]$, the terminal chance constraint is therefore equivalent to $g(y) \le 0$. Thus, at $t = 1$, the chance-constrained feasible set coincides with the clean feasible set $\mathcal{C}_1 = \{y \in \mathbb{R}^d : g(y) \le 0\}$. Consequently, $\mathcal{P}^1_{\mathrm{cc}}(x_1)$ reduces to the Euclidean projection onto $\mathcal{C}_1$, which proves the claim. □

A.3 Proof of Theorem 4.4

Proof. Since $u_\theta(\cdot, t_k)$ is L-Lipschitz, the Euler map F_k is $\gamma = 1 + hL$-Lipschitz. Indeed, for any $x, y \in \mathbb{R}^d$,

$$\begin{aligned} \|F_k(x) - F_k(y)\|_2 &= \|x - y + h(u_\theta(x, t_k) - u_\theta(y, t_k))\|_2 \\ &\le \|x - y\|_2 + h\|u_\theta(x, t_k) - u_\theta(y, t_k)\|_2 \\ &\le (1 + hL)\|x - y\|_2 = \gamma \|x - y\|_2. \end{aligned}$$

Let $d_k = \|X_k - Y_k\|_2$. Since $X_0 = Y_0$, we have $d_0 = 0$. Using $X_{k+1} = F_k(X_k) + e_k$ and $Y_{k+1} = F_k(Y_k)$, we obtain

$$\begin{aligned} d_{k+1} &= \|X_{k+1} - Y_{k+1}\|_2 \\ &= \|F_k(X_k) + e_k - F_k(Y_k)\|_2 \\ &\le \|F_k(X_k) - F_k(Y_k)\|_2 + \|e_k\|_2 \\ &\le \gamma d_k + \|e_k\|_2. \end{aligned}$$

Unrolling this recursion from $d_0 = 0$ yields

$$d_N \le \sum_{k=0}^{N-1} \gamma^{N-1-k} \|e_k\|_2,$$

which proves Equation 9.

For the Wasserstein bound, couple μ_N and μ_N^{free} by using the same initial sample $X_0 = Y_0 \sim p_0$. This is a valid coupling between the distributions of X_N and Y_N. Therefore, by the definition of W_2,

$$W_2(\mu_N, \mu_N^{\text{free}}) \leq \left(\mathbb{E}\|X_N - Y_N\|_2^2\right)^{1/2}.$$

Applying Minkowski's inequality to the pathwise bound gives

$$\begin{aligned}\left(\mathbb{E}\|X_N - Y_N\|_2^2\right)^{1/2} &\leq \left(\mathbb{E}\left[\sum_{k=0}^{N-1} \gamma^{N-1-k}\|e_k\|_2\right]^2\right)^{1/2} \\ &\leq \sum_{k=0}^{N-1} \gamma^{N-1-k}\left(\mathbb{E}\|e_k\|_2^2\right)^{1/2}.\end{aligned}$$

This proves Equation 10.

Finally, for any target distribution q with finite second moment, the triangle inequality for L^2 Wasserstein distance W_2 implies

$$W_2(\mu_N, q) \leq W_2(\mu_N, \mu_N^{\text{free}}) + W_2(\mu_N^{\text{free}}, q).$$

Combining this inequality with Equation 10 gives

$$W_2(\mu_N, q) \leq W_2(\mu_N^{\text{free}}, q) + \sum_{k=0}^{N-1} \gamma^{N-1-k}\left(\mathbb{E}\|e_k\|_2^2\right)^{1/2}.$$

□

B Implementation Details

B.1 Molecular Docking Formulation

Given a receptor with atom coordinates $\mathcal{Y} \in \mathbb{R}^{N_{\text{rec}} \times 3}$ and a ligand template conformer $\mathcal{X}_0 \in \mathbb{R}^{N_{\text{lig}} \times 3}$, we parameterize a candidate pose by a rigid-body motion $(R, t) \in \mathrm{SO}(3) \times \mathbb{R}^3$ together with a vector of rotatable-bond torsions $\phi \in \mathbb{T}^K$. The posed ligand is expressed as

$$\mathcal{X}(R, t, \phi) \;=\; R\,T_{\text{lig}}(\mathcal{X}_0; \phi) \;+\; \mathbf{1}\,\Delta^\top,$$

where $T_{\text{lig}}(\cdot; \phi)$ applies torsional rotations about designated bond axes. The goal of molecular docking is to recover ligand poses that faithfully reproduce experimentally observed binding modes while maintaining chemical validity and physical plausibility. This work considers four classes of constraints:

Bond Lengths. For each ligand bond $(i, j) \in \mathcal{E}_{\text{bond}}$,

$$(1 - \delta_b)\,\ell_{ij}^{\text{low}} \;\leq\; \|\mathcal{X}_i - \mathcal{X}_j\| \;\leq\; (1 + \delta_b)\,\ell_{ij}^{\text{up}},$$

where $\mathcal{E}_{\text{bond}}$ is the set of ligand covalent bonds, $\delta_b = 0.25$ denotes the tolerance, $\ell_{ij}^{\text{low}}, \ell_{ij}^{\text{up}}$ are chemically derived lower/upper bounds (in Å) for bond (i, j), $\mathcal{X}_i \in \mathbb{R}^3$ and $\mathcal{X}_j \in \mathbb{R}^3$ denote the Cartesian coordinates (in Å) of the i-th and j-th ligand atom, respectively.

Bond Angles. For every bond-angle triplet $(i{-}j{-}k) \in \mathcal{E}_\angle$,

$$(1 - \delta_\alpha)\,\alpha_{ijk}^{\text{low}} \;\leq\; \angle(i{-}j{-}k) \;\leq\; (1 + \delta_\alpha)\,\alpha_{ijk}^{\text{up}},$$

where $\mathcal{E}_\angle$ denotes the set of bond-angle triplets, $\angle(i{-}j{-}k)$ is the geometric angle at vertex j, $\alpha_{ijk}^{\text{low}}, \alpha_{ijk}^{\text{up}}$ are lower/upper bounds (in radians) for angle $(i{-}j{-}k)$, $\delta_\alpha = 0.25$ denotes the bond angles tolerance.

Internal Steric Clash (Intra-ligand). For all ligand atom pairs (i, j) not directly bonded or forming an angle, i.e., $(i, j) \notin \mathcal{E}_{\text{bond}} \cup \mathcal{E}_{\angle}$,

$$\|\mathcal{X}_i - \mathcal{X}_j\| \;\geq\; (1 - \delta_{\text{steric}})\, d_{ij}^{\text{low}},$$

where d_{ij}^{low} is the lower bound (in Å) on permissible separation for nonbonded, nonangle ligand pairs, $\delta_{\text{steric}} = 0.2$ is the internal steric clash tolerance.

Minimum Protein–Ligand Contact. Defining the closest cross-molecule distance as

$$d_{\min}(\mathcal{X}, \mathcal{Y}) \;=\; \min_{i \in \text{lig},\, j \in \text{rec}} \|\mathcal{X}_i - \mathcal{Y}_j\|,$$

We require that the van der Waals overlap does not exceed a tolerance $\delta_{\text{pl}} = 0.75$Å:

$$\forall (i, j) :\; d_{lp} \;\geq\; (\rho_i + \rho_j) - \delta_{\text{pl}}, \quad \text{equivalently} \;\; \max_{i,j} \max\big(0,\, \rho_i + \rho_j - d_{ij}\big) \;<\; \delta_{\text{pl}},$$

where d_{ij} is the distance between ligand atom i and protein atom j, and ρ_i, ρ_j denote their van der Waals radii. Second, to ensure the ligand is not placed too far from the binding pocket, we require the existence of at least one contact.

B.2 PDE Problem Formulation

B.2.1 Reaction–Diffusion Equation

We consider the one-dimensional nonlinear reaction–diffusion equation

$$\frac{\partial v}{\partial t} \;=\; \nu \frac{\partial^2 v}{\partial s^2} + \rho\, v(1 - v),$$

where $v(s, t)$ denotes the state of the species at position s and time t, $\nu > 0$ is the diffusion coefficient, and $\rho > 0$ is the reaction rate constant.

The solution $v(s, t)$ must satisfy the following two constraints:

Initial Condition.

$$-\delta_{\text{pde}} \;\leq\; v(s, 0) - v_{\text{IC}}(s) \;\leq\; \delta_{\text{pde}},$$

where $v_{\text{IC}}(s)$ denotes the initial state, $\delta_{\text{pde}} = 1 \times 10^{-13}$ is the tolerance. This condition ensures that the solution starts from a specific initial condition $v_{\text{IC}}(s)$ at time $t = 0$.

Nonlinear Mass Conservation.

$$-\delta_{\text{pde}} \;\leq\; m(t) - \left(m(0) \;+\; \int_0^t \rho\nu(1 - \nu) dt \;+\; \int_0^t (g_L - g_R)\, dt \right) \;\leq\; \delta_{\text{pde}},$$

where $m(t) = \int v(s, t)\, ds$ is the total mass at time t, and g_L, g_R are the Neumann boundary fluxes at the left and right boundaries, respectively.

These constraints ensure consistency with the prescribed initial condition and the correct nonlinear evolution of the total mass due to reaction and boundary effects.

B.2.2 Navier—Stokes Equation

The dynamics of incompressible fluids are governed by the Navier–Stokes equations:

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla \pi + \nu \nabla^2 \mathbf{v} + \mathbf{f}, \qquad \nabla \cdot \mathbf{v} = 0,$$

where $\mathbf{v}(s, t)$ is the velocity field, $\pi(s, t)$ is the pressure field, $\nu > 0$ is the kinematic viscosity, $\mathbf{f}(s, t)$ is an external force (e.g., gravity).

The solution must satisfy the following two constraints:

Initial Condition.

$$-\delta_{\text{pde}} \;\le\; \mathbf{v}(s,0) - \mathbf{v}_{\text{IC}}(s) \;\le\; \delta_{\text{pde}},$$

where $\mathbf{v}_{\text{IC}}(s)$ is the initial state. This condition ensures the velocity field starts from a prescribed initial condition.

Global Mass Conservation.

$$-\delta_{\text{pde}} \;\le\; \iint \mathbf{v}(s,t)\,ds \;-\; \iint \mathbf{v}(s,0)\,ds, \;\le\; \delta_{\text{pde}},$$

ensuring that the total mass of the incompressible fluid remains constant over time.

B.3 Physics-Informed Motion Formulation

We model the vertical motion of a rigid object falling under uniform gravity. Each trajectory contains six frames corresponding to a physical duration of 4 seconds. The horizontal location is fixed, and only the vertical coordinate evolves according to the gravitational dynamics.

Free-Fall Dynamics. Let $\mathbf{p}_t \in \mathbb{R}^2$ denote the object position at frame t, and let $\mathbf{v}_t$ denote its velocity. The motion obeys the discrete-time update rule

$$\mathbf{p}_t = \mathbf{p}_{t-1} + \left(\mathbf{v}_t + 0.5\,\frac{\partial \mathbf{v}_t}{\partial t}\right), \tag{12a}$$

$$\mathbf{v}_{t+1} = \frac{\partial \mathbf{p}_t}{\partial t} + \frac{\partial \mathbf{v}_t}{\partial t}, \tag{12b}$$

where $\frac{\partial \mathbf{v}_t}{\partial t}$ is the gravitational acceleration. During training we use Earth gravity

$$\frac{\partial \mathbf{v}_t}{\partial t} \;=\; \mathbf{a}_{\text{E}} = (0,\, -9.8),$$

and during evaluation we impose Moon gravity

$$\frac{\partial \mathbf{v}_t}{\partial t} \;=\; \mathbf{a}_{\text{M}} = (0,\, -1.62).$$

We use 2048 trajectories for training and 205 for validation. For evaluation, we generate 100 trajectories under Moon gravity using the three methods (unconstrained generation, repeated projection, and CCFM).

B.4 Constraints Enforcement

B.4.1 Molecular Docking

To enforce geometric feasibility on molecular conformations, we adopt an iterative, gradient-free projection strategy. Starting from an initial state $x^{(0)}$, the algorithm generates a sequence of iterates $x^{(k+1)} = \text{Project}(x^{(k)})$, where each projection step reduces the violation of geometric constraints. The process terminates once no constraint is violated or a fixed iteration budget is reached, yielding a final feasible conformation.

At each step, we first identify the active set of violated constraints, denoted $\mathcal{A}(x)$. For each violated constraint, we compute a corrective displacement along the axis connecting the two atoms involved. The displacement magnitude is proportional to the violation and scaled by a small fixed step size, producing a push or pull effect that nudges atoms toward feasibility. The updates across all active constraints are aggregated into a single correction Δx, with per-atom displacements capped to ensure numerical stability. The coordinates are then updated as $x \leftarrow x + \Delta x$.

While a single update reduces constraint violations, it may not achieve strict feasibility. Iterative application guarantees convergence to a geometrically valid conformation. By restricting updates to the active set, the method avoids unnecessary corrections on already satisfied constraints, while still ensuring that all geometric conditions are satisfied at convergence.

B.4.2 PDE

Following the enforcement strategy of [48], we also adopt the Gauss–Newton method for enforcing constraints in PDE formulations. For linear constraints, the projection is exact and guarantees feasibility. For nonlinear constraints, the single projection provides only an approximate correction, and strict feasibility is deferred to the final step, where multiple Gauss–Newton iterations are applied. In addition, since our setting considers inequality constraints, we adopt an *active set strategy*. Specifically, at each step we identify the subset of constraints that are currently violated and apply projection updates only to this active set. By restricting correction to infeasible constraints, this approach avoids unnecessary computations on already satisfied constraints, while still ensuring that all constraints are satisfied at convergence.

It is important to highlight that, although all constraint enforcement is based on the Gauss–Newton method, our approach differs from PCFM. In PCFM, projections are repeatedly applied to the original constraints defined on the clean sample, regardless of sampling time. In contrast, our method introduces a *chance-constrained projection operator*, which adaptively adjusts the tightness of constraints according to the sampling time. This adaptive mechanism is supported by theoretical guarantees, ensuring both efficiency and correctness.

Linear Projection For linear inequality constraints, projection is exact. At each sampling step, we solve

$$x_{\text{proj}} \;=\; \arg\min_{z} \|z - x\|_2^2 \quad \text{s.t.} \quad g(z) \le 0,$$

which admits a closed-form solution due to the rank-1 structure of the Jacobian. For instance, in the Navier—Stokes Equation, the total-vorticity constraint reduces to a constant-shift correction

$$\mathbf{v}(\cdot,\cdot,k) \;\leftarrow\; \mathbf{v}(\cdot,\cdot,k) - \text{average residual}.$$

This closed-form update yields exact feasibility at negligible cost and is applied after every sampling step.

Gauss–Newton Projection For general nonlinear constraints, we employ an approximate correction. Given a candidate $x \in \mathbb{R}^n$ subject to $g_i(x) \le 0$, we first identify the active set of violated constraints

$$\mathcal{A}(x) = \{\, i \mid g_i(x) > 0 \,\},$$

with residuals and Jacobian

$$r = g_{\mathcal{A}(x)}(x), \qquad J = \nabla g_{\mathcal{A}(x)}(x)^\top \in \mathbb{R}^{n \times |\mathcal{A}(x)|}.$$

A single Gauss–Newton update

$$x_{\text{proj}} = x - J^\top (JJ^\top)^{-1} r$$

reduces the residuals of the active constraints, pushing the iterate back toward the feasible region. In PCFM and CCFM, this step is applied after each sampling iteration.

Specifically, we implement the active set via hinge residuals $r = \mathrm{ReLU}(g(x))$ for efficient GPU batching. The linear system is stabilized by adding the regularization

$$y = (JJ^\top + \lambda I)^{-1} r, \quad \Delta x = -J^\top y, \quad x \leftarrow x + \Delta x,$$

with $\lambda \approx 10^{-6}$.

Final Projection for Strict Feasibility To guarantee exact satisfaction of all constraints in the generated samples, we perform Newton–Schur iterations restricted to the active set:

$$x^{(k+1)} = x^{(k)} - J_k^\top (J_k J_k^\top)^{-1} g_{\mathcal{A}(x^{(k)})}\big(x^{(k)}\big), \quad J_k = \nabla g_{\mathcal{A}(x^{(k)})}(x^{(k)})^\top.$$

Under smoothness and full-rank assumptions, these iterations converge quadratically [48]. In both PCFM and CCFM, we allocate a fixed budget of 30 iterations after the final sampling step to ensure strict feasibility.

B.5 Dataset Details

B.5.1 Molecular Docking

This work utilizes the same dataset as adopted in [11], namely the PDBBind benchmark [33], which has become a standard for evaluating docking models. Following previous work [47, 11], we employ the time-based split, where approximately 17,000 complexes released before 2019 are used for training and validation, while the 363 complexes released after 2019 constitute the test set.

To ensure consistency between apo and holo structures, we first sanitize the raw PDBBind files using PDBFixer [12] to replace non-standard residues and add missing heavy atoms. For the apo state, we extract protein sequences from the processed files and generate structural predictions with ESMFold [28]. These predictions are further refined by PDBFixer to add missing terminal atoms, while the crystallographic bound conformations from PDBBind serve as the holo state. Finally, hydrogen atoms are removed during alignment to reduce inconsistencies arising from variable hydrogen placement.

B.5.2 PDE

Following the dataset construction strategies in [7, 48], we generate training and testing trajectories by combining randomized initial conditions with varying boundary or forcing configurations. The key distinction is that, whereas prior works typically employ only 10 initial conditions in the test set, we adopt 90. This design probes model performance under more diverse and complex scenarios, thereby yielding a more challenging and discriminative evaluation.

Reaction–Diffusion Equation The 1D reaction–diffusion system is simulated on the spatial domain $s \in [0, 1]$ with $n_s = 128$ grid points and over $n_t = 100$ discrete time steps. The numerical simulations were performed using FiPy [16], a Python-based finite volume solver for partial differential equations. We employ a semi-implicit scheme with a fixed diffusion coefficient and reaction rate of $(\nu, \rho) = (0.005, 0.01)$.

The initial conditions are drawn from randomized combinations of sinusoidal modes and localized perturbation functions, providing both smooth and sharp spatial profiles. Neumann boundary conditions are enforced with time-varying fluxes (g_L, g_R) sampled from a prescribed range.

The dataset is split into training and testing sets. To form the training set, we pair 100 distinct initial conditions with 100 unique boundary flux configurations, yielding a total of 10,000 solution trajectories. The test set consists of 1,000 trajectories generated using initial conditions and boundary flux configurations distinct from those used for training. The conditions used for the training and test sets are entirely disjoint.

Navier—Stokes Equation We consider the 2D incompressible Navier–Stokes equations in vorticity form on a periodic square domain discretized with a 64×64 spatial grid. Temporal evolution is simulated over $T = 49$ time units, with 50 uniformly sampled snapshots recorded per trajectory. We adopt a Crank–Nicolson spectral solver to ensure stability for the viscosity parameter $\nu = 10^{-3}$. The initial vorticity fields are sampled from Gaussian random fields with prescribed covariance structure, while the external forcing is defined as

$$f(x) \;=\; \tfrac{0.1}{\sqrt{2}}\,\sin\big(2\pi(x_1 + x_2) + \varphi\big), \qquad \varphi \sim \mathcal{U}(0, \pi/2).$$

To construct the training dataset, we generate 10,000 trajectories by sampling 100 random initial conditions and 100 random forcing phases. The test set consists of 1,000 randomly selected trajectories generated using initial conditions and forcing phases distinct from those used for training. The initial conditions used for training and evaluation are disjoint.

B.6 Model Details

B.6.1 Molecular Docking

For our experiments, we adopt the FlexDock framework, a state-of-the-art generative model for flexible molecular docking based on Unbalanced Flow Matching (UFM) [11]. FlexDock integrates two complementary modules: a manifold docking flow, which operates in reduced degrees of freedom

to efficiently capture ligand rigid-body motions and protein side-chain flexibility, and a subsequent relaxation flow, which refines atomic coordinates in Euclidean space under energy-based constraints to ensure physically valid conformations.

This work directly utilizes the official pre-trained FlexDock models provided by the authors, trained on the PDBBind dataset with time-based splits. This setup allows us to leverage a robust, well-validated backbone without additional fine-tuning, thereby ensuring reproducibility, comparability with prior benchmarks, and fairness in experimental comparisons.

B.6.2 PDE

For all PDE benchmarks, we adopt the Functional Flow Matching (FFM) framework parameterized by a Fourier Neural Operator (FNO) backbone [23]. The FNO encoder processes the concatenation of the current state, a sinusoidal positional embedding of the spatial grid, and a Fourier time embedding, thereby enabling spatiotemporal generalization across functional domains. The resulting time-dependent vector field defines the flow dynamics used in all experiments. Table 5 and Table 6 summarize the hyperparameters used during training for reaction–Diffusion and Navier–Stokes equations, respectively.

Table 5: Params for Reaction–Diffusion.

Hyperparameter	Value
Fourier Layers	6
Fourier Modes	[32,32]
Hidden Channels	64
Projection Channels	256
Batch Size	256
Optimizer	adam
Scheduler	plateau
Prior Distribution	randn
Training Steps	20000

Table 6: Params for Navier–Stokes.

Hyperparameter	Value
Fourier Layers	4
Fourier Modes	[16,16,16]
Hidden Channels	64
Projection Channels	256
Batch Size	16
Optimizer	adamw
Scheduler	plateau
Prior Distribution	randn
Training Steps	200000

B.7 Sampling Setup

B.7.1 Molecular Docking

FlexDock employs two distinct flow-matching models for molecular docking: one dedicated to docking and the other to relaxation. Constraint enforcement is primarily handled by the second model, which applies an Euler-based update scheme. At each inference step, multiple candidate samples are generated, and a discriminator is subsequently used to select the single most promising configuration as the final output. We evaluate FlexDock under varying sample counts (1, 5, 10, 15, 20) and inference steps (2, 5, 10, 15, 20), reporting the detailed results in Section 5 and the remaining results in the appendix C.

CCFM adopts a sampling setup largely consistent with FlexDock, employing the same ranges of sample counts and inference steps. The only distinction is that CCFM enforces constraints after each inference step using a gradient-free method shown in Appendix B.4, followed by a final refinement of 20 iterations once the sampling trajectory is complete.

B.7.2 PDE

For all constraint-aware methods, we employ the modified Euler scheme with 200 sampling steps to ensure a fair comparison.

CCFM (ours). Our method performs explicit Euler integration with constraint correction at every step. Each intermediate state is projected onto the constraint manifold using a single Newton update. As shown in Proposition 4.2, the deterministic reformulation of chance constraints is governed by both the variance of the induced random variable and the probability of constraint satisfaction. Specifically, the variance depends on the sampling time t, while the satisfaction probability is given by $\phi_n(t) = \frac{1}{2}t^n$. The parameter n thus controls the tightening schedule: larger values (e.g., $n = 0.5$)

Table 7: Normalized deviations from straight interpolation between the initial noise and clean sample.

Mean	Median	95th percentile
0.8622%	0.8544%	1.2987%

Table 8: Sensitivity analysis for probability of constraint satisfaction. The probability is determined by the function $\phi_n(t) = \frac{1}{2}t^n$, and we adjust n to observe the results. Results are reported using #S=1 and #Stp=10. PB Valid (+L < 2Å or +A < 2Å) denotes the percentage of physically valid samples that also satisfy the respective RMSD thresholds.

Metric	**CCFM(0.7)**	**CCFM(0.9)**	**CCFM(1.0)**	**CCFM(1.1)**	**CCFM(1.3)**
PB Valid (**+L** < 2**Å**)	29.85	30.16	30.79	30.48	30.48
PB Valid (**+A** < 2**Å**)	58.19	57.85	58.19	58.19	58.88

delay tightening, while smaller values (e.g., $n = 0.1$) apply stronger corrections earlier. We adopt $n = 0.5$ for the reaction–diffusion task and $n = 0.1$ for the Navier–Stokes task. A detailed sensitivity analysis with respect to n is provided in the Appendix C. After the sampling process, we apply 30 iterations of Gauss–Newton refinement to guarantee the strict feasibility of all constraints.

PCFM. This baseline relies on iterative extrapolation–correction–interpolation steps combined with periodic mixing and noise resampling. To align with our fixed sampling budget, we restrict the number of mixing steps to 2. Constraint correction follows the same projection-based strategy as in CCFM.

Diffusion PDE. We follow the gradient-guided sampling procedure [18] on a pretrained FFM backbone. Constraint satisfaction is encouraged by augmenting the generation loss with explicit penalties on both the initial condition and the conservation law. The composite loss is weighted by a factor of 200 to balance fidelity and feasibility.

B.8 Hardware

For the molecular docking experiments, inference was conducted on six NVIDIA A6000 GPUs. For the Reaction–Diffusion, both training and inference were performed on a single NVIDIA A100 GPU. For the Navier–Stokes, the training process was done on six NVIDIA A6000 GPUs, while inference was performed on a single NVIDIA A100 GPU.

C Additional Results

C.1 Marginal Trajectory Curvature

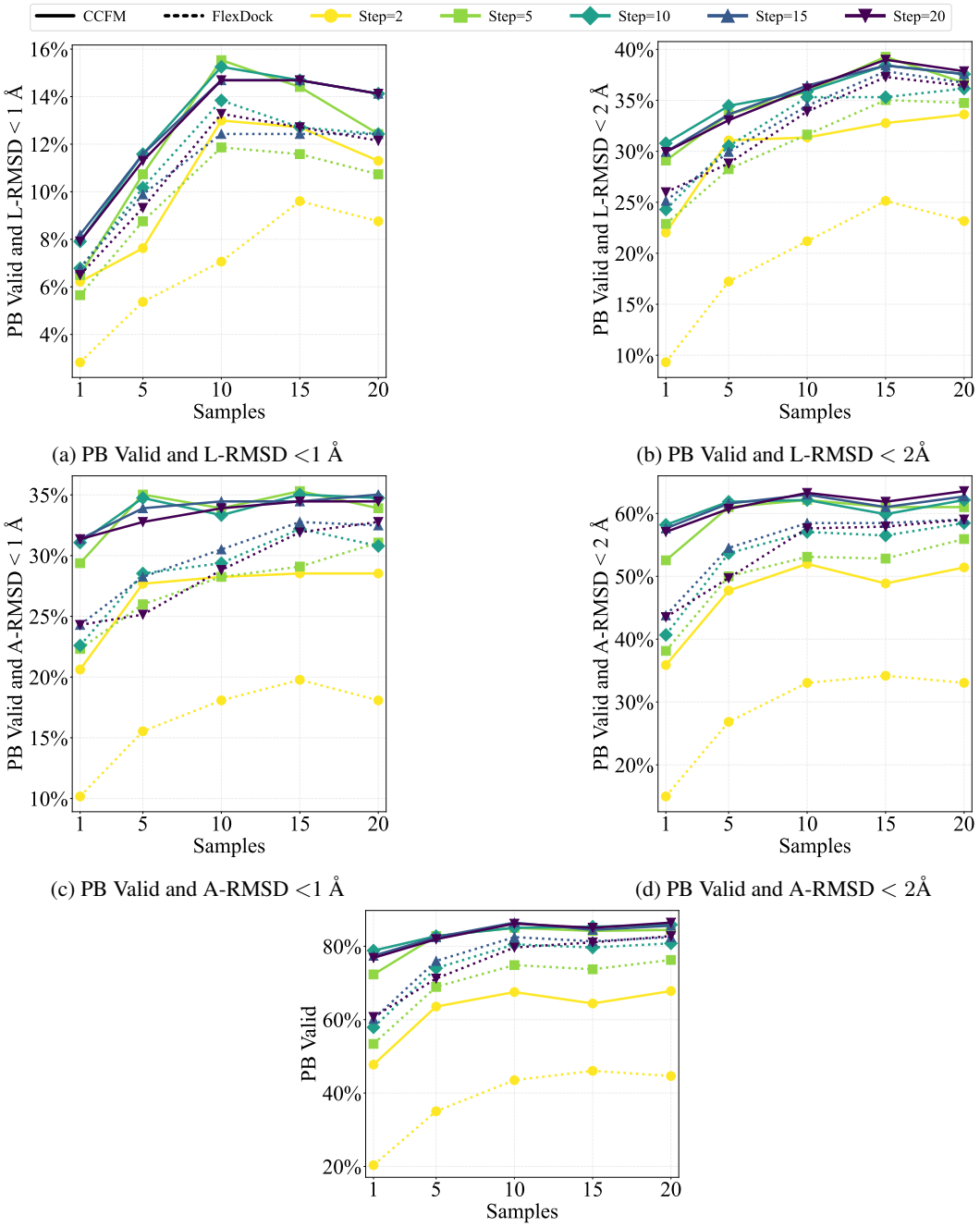
To quantify the deviation of learned marginal sampling trajectories from the straight conditional interpolation in Equation 1, we evaluate 1000 unconstrained Reaction–Diffusion sampling trajectories. Let x_0 be the initial noise and x_1 the clean sample. We compare each intermediate state x_t with the straight interpolation $\ell_t = (1-t)x_0 + tx_1$ using

$$d_t = \frac{\|x_t - \ell_t\|_2}{\|x_1 - x_0\|_2}. \tag{11}$$

The deviation is normalized by the distance between the initial noise and clean sample. Table 7 shows that the mean normalized deviation from straight interpolation between the initial noise and clean sample is below 1%. These measurements provide empirical support for the straight-line approximation.

C.2 Molecular Docking

A comparison between CCFM and FlexDock across varying sample sizes and inference steps is shown in Figure 5.



(a) PB Valid and L-RMSD <1 Å

(b) PB Valid and L-RMSD < 2Å

(c) PB Valid and A-RMSD <1 Å

(d) PB Valid and A-RMSD < 2Å

(e) PB Valid

Figure 5: Comparison of CCFM (solid lines) and FlexDock (dashed lines) under different sampling budgets and evaluation metrics.

At the strictest setting (PB Valid with L-RMSD < 1 Å, step = 2, sample = 1), CCFM attains over 6% success rate, more than doubling FlexDock's 3%. A similar advantage persists when relaxing the cutoff to L-RMSD < 2 Å, where CCFM consistently surpasses FlexDock across all sample and step budgets.

When evaluated on A-RMSD, the margin becomes even more pronounced: for PB Valid with A-RMSD < 1 Å, CCFM exceeds FlexDock by at least 5% in nearly every configuration, and maintains comparable superiority at the A-RMSD < 2 Å threshold.

Table 9: PDBBind docking results with standard deviation (#S=1, #Stp=10).

Methods	L-RMSD (%)		A-RMSD (%)	PB Valid (%)		
	< 1Å	< 2Å	< 2Å	All	+L < 2Å	+A < 2Å
CCFM	$7.80 \pm (0.31)$	$32.57 \pm (0.57)$	$72.27 \pm (0.69)$	$79.15 \pm (0.41)$	$30.45 \pm (0.68)$	$58.53 \pm (0.49)$
FlexDock	$7.45 \pm (0.43)$	$31.92 \pm (0.81)$	$72.65 \pm (0.81)$	$58.07 \pm (1.24)$	$24.57 \pm (1.13)$	$40.96 \pm (1.38)$
RP	$7.80 \pm (0.34)$	$32.19 \pm (0.79)$	$72.26 \pm (0.99)$	$56.48 \pm (0.72)$	$27.68 \pm (0.81)$	$42.93 \pm (0.85)$

Table 10: Comparison of running time between CCFM and PCFM on Reaction–Diffusion and Navier–Stokes equations.

Problem	Metric	CCFM (ours)	PCFM
Reaction–Diffusion	Running Time (s) (↓)	2.703	3.874
Navier–Stokes	Running Time (s) (↓)	3.065	3.077

For PB Valid, with a moderate budget (5 samples, 5 steps), CCFM achieves a PB Valid level comparable to FlexDock with 20 samples and 20 steps, reducing the effective sample-step cost **from 400 to just 25**. These results highlight the overall trend: CCFM consistently matches or exceeds FlexDock validity within a fraction of the computational expense.

The improvements are most striking under limited budgets, where CCFM achieves strong validity and fidelity with far fewer samples and steps. Even as the computational budget increases, CCFM continues to deliver higher feasibility and fidelity while maintaining a clear advantage in efficiency.

Figure 6 illustrates representative docking results on six PDBBind test complexes, highlighting the alignment between predicted and ground-truth protein–ligand structures.

Table 8 shows that varying the probability scheduler parameter n has a negligible impact on performance. This demonstrates the method's robustness to the choice of scheduler.

Table 9 presents the mean $\pm$ standard deviation results from 5 independent experimental runs, validating docking methods on the PDBBind dataset. CCFM consistently demonstrates superior performance and stability compared to FlexDock and RP.

C.3 PDE solution generation

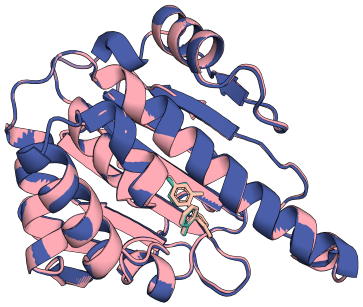
We report additional results of the PDE solution generation tasks. This section presents the running time for different methods, sensitivity analysis of the probability scheduler, MMSE curves of the generated solution fields varying with physical time, the visualization of SMSE and MMSE with all time frames (Navier–Stokes), the pointwise average residuals varying with physical time, and the comparison of models using an example of the generated solution field.

Table 10 shows that CCFM runs faster than PCFM while still ensuring feasibility. The gain is most pronounced in Reaction–Diffusion, where nonlinear projections slow down PCFM. By adaptively relaxing constraints with the chance-constrained operator, CCFM often has smaller active corrections due to relaxation, which avoids unnecessary projections and reduces computational cost.

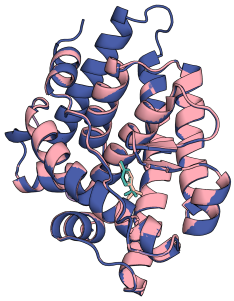
C.3.1 Reaction–Diffusion

Table 11 presents the sensitivity analysis of the probability scheduler hyperparameter. We observe that varying the parameter n has almost no impact on constraint satisfaction, consistently ensuring feasibility across all settings. The effects on MMSE and SMSE are also marginal, with only slight fluctuations that remain within a reasonable range. This indicates that the method is robust to the choice of the probability scheduler.

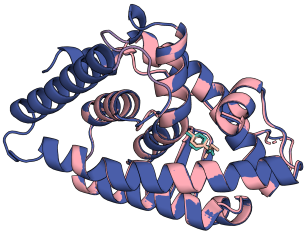
Figure 7 shows the MMSE curves as a function of physical time for the Reaction–Diffusion equation. CCFM demonstrates superior performance compared with the other models used in this study, while exactly satisfying the initial condition (with error equal to 0). PCFM has similar performance. Figure 8 shows the SMSE, where the x-axis represents physical time and the y-axis represents space. Similar to the MMSE (Figure 3), the proposed CCFM achieves the lowest errors compared with other models. The example solution fields in Figure 9 demonstrates the improvement from CCFM compared with other models in generating a specific solution field. The CCFM-generated sample



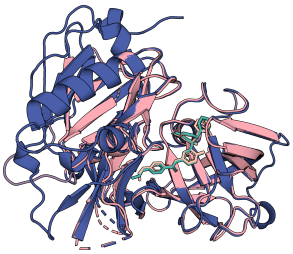
(a) 6hhr



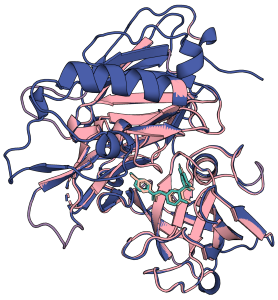
(b) 6i65



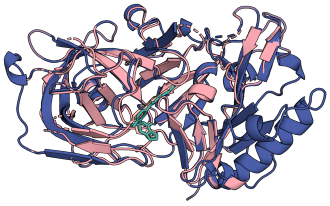
(c) 6i66



(d) 6jse



(e) 6jsg



(f) 6od6

Figure 6: Visualization for the generated complexes(6HHR, 6I65, 6I66, 6JSE, 6JSG, and 6OD6) from the PDBBind test dataset. Predicted protein structures are shown in pink, predicted ligands in teal, ground-truth proteins in dark blue, and ground-truth ligands in melon.

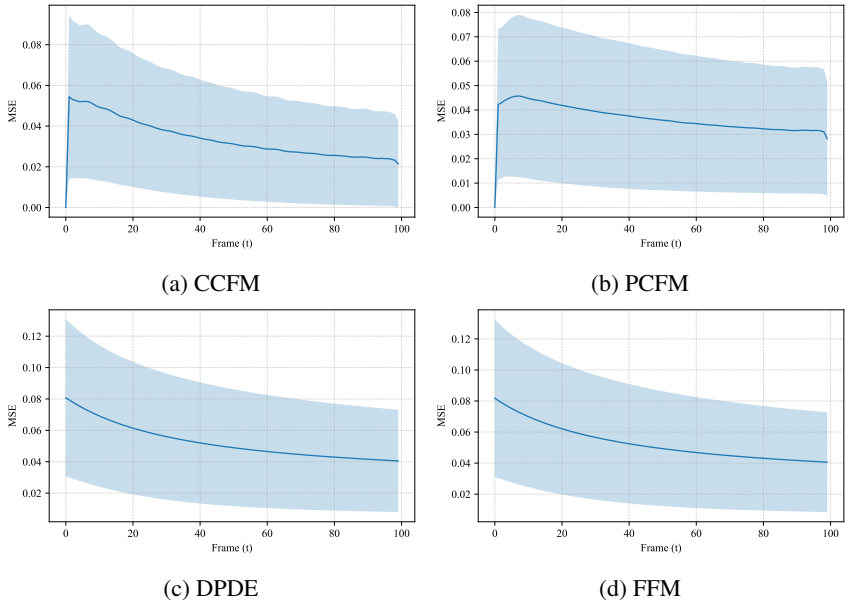
closely adheres to the ground truth, while all other models show substantial discrepancies. These additional results consistently highlight the superiority of the proposed CCFM in generating solutions with high accuracy while ensuring constraint satisfaction.

C.3.2 Navier–Stokes

For the Navier–Stokes problem, to complement the results presented in Section 5.4, we report evaluation results for all physical time frames. Figure 11 shows the MMSE of generated samples from the models as a function of physical time. While the errors increase over time, CCFM maintains relatively low error levels across all frames compared with other models. This is also reflected in Figure 10, where CCFM outperforms other models, with the maximum spatially averaged MMSE around 0.12 (others $>$ 0.20). Meanwhile, the average solution residuals from CCFM, shown in

Table 11: Sensitivity analysis for probability of constraint satisfaction. The probability is determined by the function $\phi_n(t) = \frac{1}{2}t^n$, and we adjust n to observe the results.

Problem	Metric	CCFM (0.1)	CCFM (0.3)	CCFM (0.5)	CCFM (0.7)	CCFM (0.9)
Reaction–Diffusion	MMSE $\times 10^{-2}$ (↓)	3.6	3.6	3.3	3.2	3.2
	SMSE $\times 10^{-2}$ (↓)	2.8	2.9	2.7	2.6	2.6
	CV (IC) $\times 10^{-2}$ (↓)	0	0	0	0	0
	CV (CL) $\times 10^{-15}$ (↓)	9.9	9.6	9.5	9.5	9.5



(a) CCFM (b) PCFM

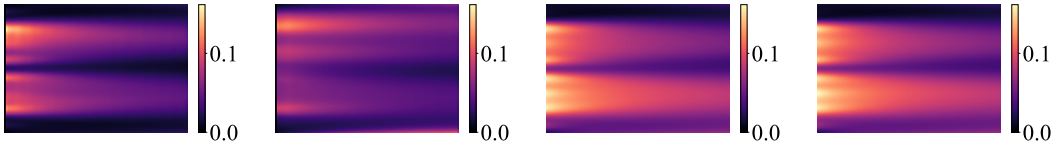
(c) DPDE (d) FFM

Figure 7: Comparison of generated solutions for the Reaction–Diffusion equation in terms of MMSE varying with physical time.

Figure 13, are significantly lower than those of other models across all time frames. In all cases, CCFM not only produces solutions with high accuracy but also yields samples that strictly satisfy the constraints. Figure 14 shows an example of the true solution field compared with generated ones from the models. Visually, the CCFM-generated sample consistently aligns with the ground truth, especially at early time steps where DPDE and FFM fail. PCFM achieves closer performance in the early steps yet exhibits larger magnitude discrepancies (lighter blue than the ground truth). Table 12 presents a comprehensive performance comparison of five different methods on the 2D Navier–Stokes equation solution generation task. Crucially, all results are reported as the mean performance with associated standard deviations, derived from 5 independent runs, demonstrating that the CCFM method achieves the best overall and most stable performance across the key error and conservation metrics.

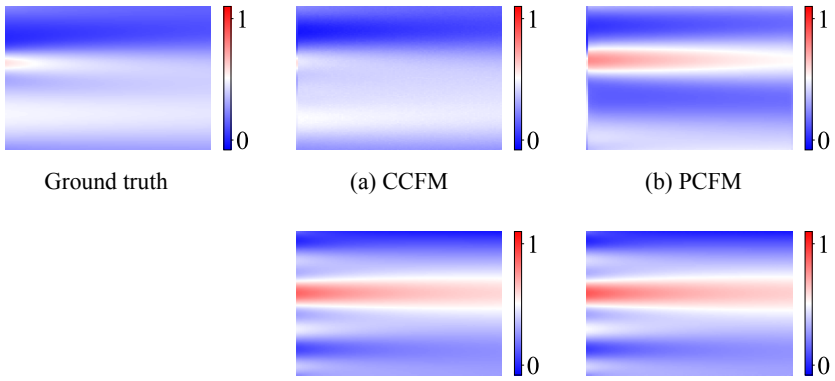
C.4 Physics-Informed Motion

Figure 15 illustrates generated trajectories under Moon gravity, where the proposed approaches align closely with the ground truth (red circles) compared to the baseline while ensuring high-fidelity generated samples.



(a) CCFM (b) PCFM (c) DPDE (d) FFM

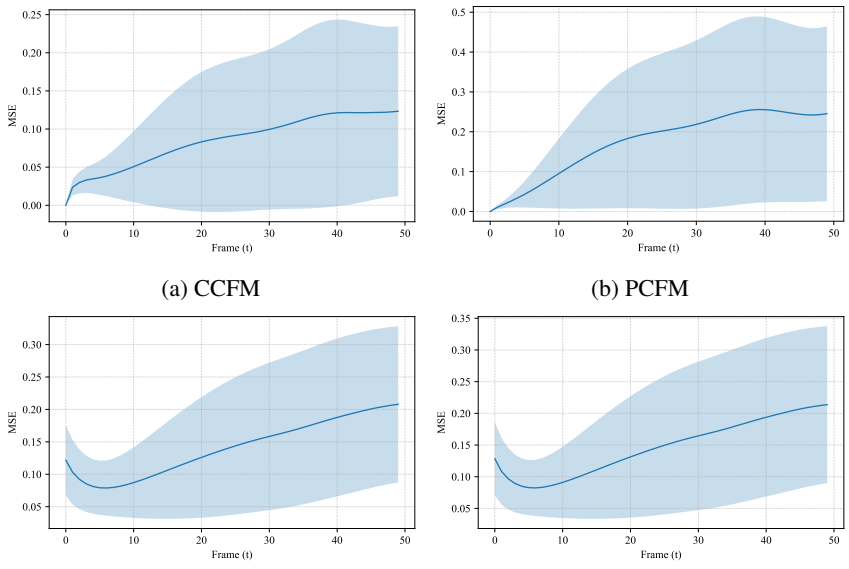
Figure 8: SMSE of Reaction–Diffusion solutions over physical time, with the horizontal axis representing time evolution and the vertical axis indicating the state at each time step.



Ground truth (a) CCFM (b) PCFM

(c) DPDE (d) FFM

Figure 9: Example solution fields of the Reaction–Diffusion equation generated using the models compared with the ground truth.



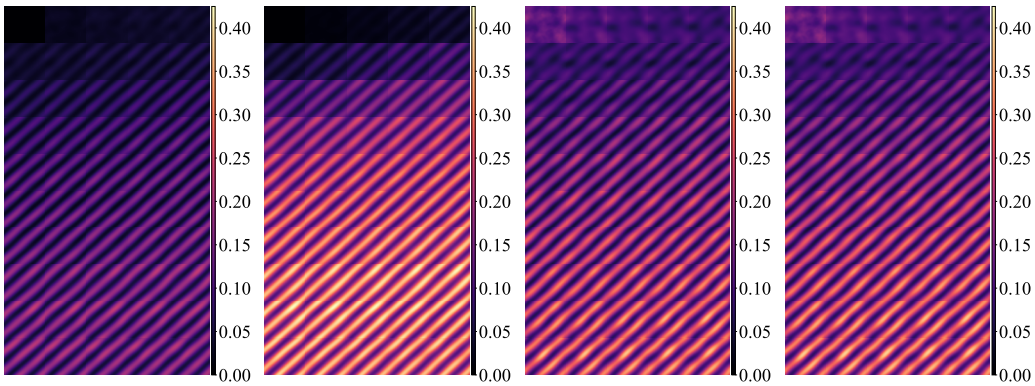
(a) CCFM (b) PCFM

(c) DPDE (d) FFM

Figure 10: Comparison of generated solutions for the Navier–Stokes equation in terms of MMSE varying with physical time.

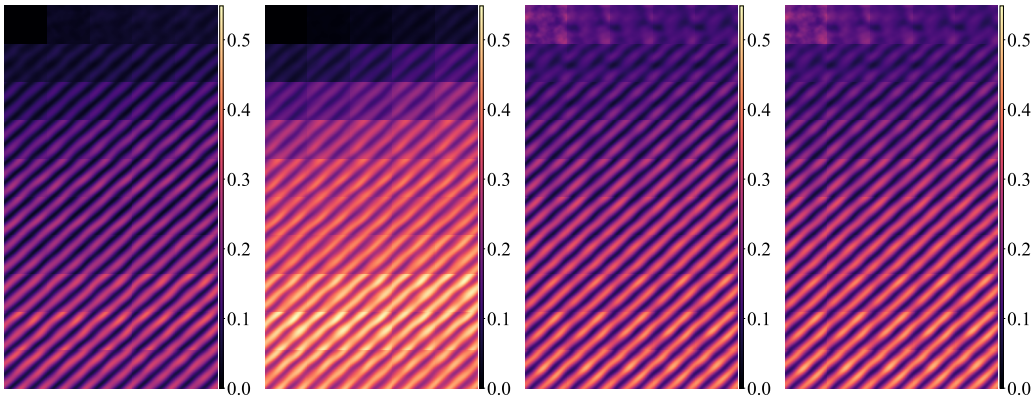
Table 12: Results with standard deviations for solution generation on the 2D Navier–Stokes equation. A lower value indicates better performance. Bold indicates the best value in each row.

Problem	**Metric**	**CCFM (ours)**	**PCFM**	**DiffusionPDE**	**FFM**	**RP**
Navier–Stokes	MMSE $\times 10^{-1}$ (↓)	$0.79 \pm (0.03)$	$1.67 \pm (0.09)$	$1.44 \pm (0.10)$	$1.56 \pm (0.16)$	$1.08 \pm (0.08)$
	SMSE $\times 10^{-1}$ (↓)	$0.81 \pm (0.11)$	$1.53 \pm (0.18)$	$9.21 \pm (0.39)$	$9.57 \pm (0.53)$	$0.97 \pm (0.25)$
	CV (IC) (↓)	0	0	$0.12 \pm (0.03)$	$0.13 \pm (0.08)$	0
	CV (CL) $\times 10^{-15}$ (↓)	$3.9 \pm (0.01)$	$3.4 \pm (0.02)$	$1.2 \pm (0.13) \times 10^{10}$	$1.2 \pm (0.12) \times 10^{9}$	$4.1 \pm (0.04)$



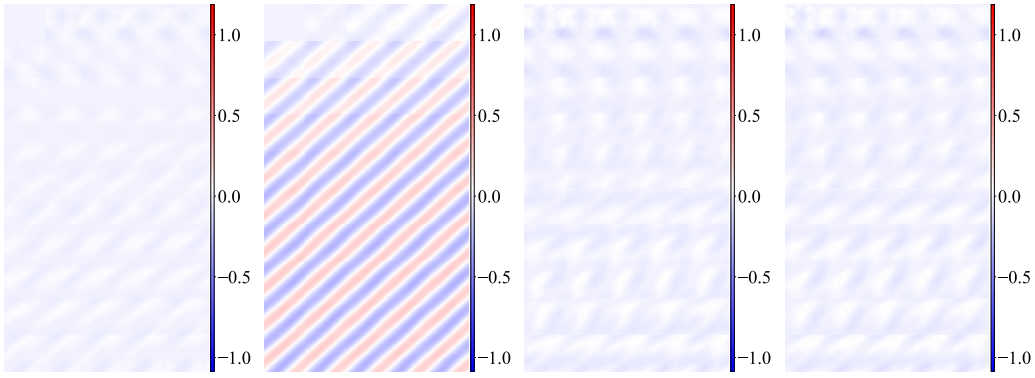
(a) CCFM (b) PCFM (c) DPDE (d) FFM

Figure 11: MMSE (the darker the better) of Navier–Stokes solutions over physical time with all 50 frames. Each frame corresponds to the system state at a given physical time.



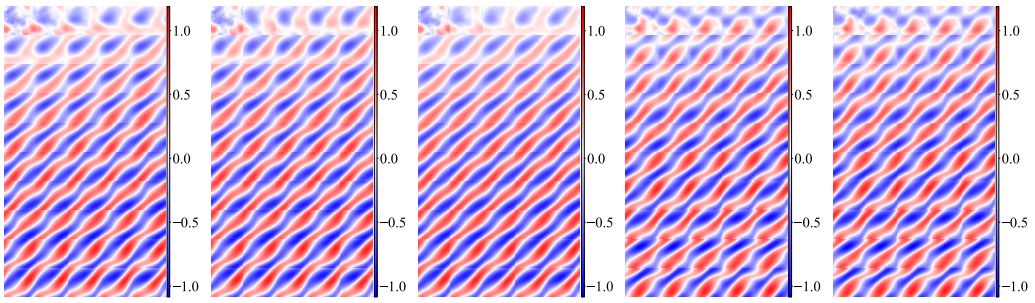
(a) CCFM (b) PCFM (c) DPDE (d) FFM

Figure 12: SMSE (the darker the better) of Navier–Stokes solutions over physical time with all 50 frames. Each frame corresponds to the system state at a given physical time.



(a) CCFM (b) PCFM (c) DPDE (d) FFM

Figure 13: Point-wise averaged residuals (the lighter the better) of Navier–Stokes solutions over physical time with all 50 frames. Each frame corresponds to the system state at a given physical time.



(a) Ground truth (b) CCFM (c) PCFM (d) DPDE (e) FFM

Figure 14: Example solution fields of the Navier–Stokes equation generated using the models compared with the ground truth.

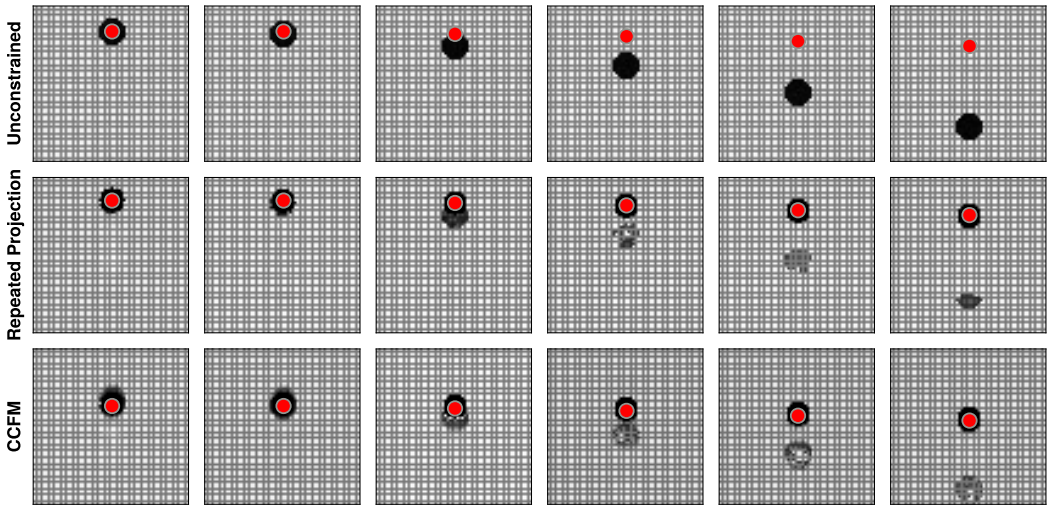


Figure 15: Generated trajectories instances for the physics-informed motion generation. In each frame, the black solid objects are generated using three constraint-handling approaches. The red circles represent the correct locations of objects under Moon gravity.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The results in experiments support the claims made in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The paper includes formal theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper describes the benchmark setting, baseline methods, evaluation metrics, and key implementation details needed to reproduce the main experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code in the supplementary materials.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the benchmark settings, baselines, evaluation metrics, and key algorithmic parameters used in the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the statistical significance in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the hardware setup and the compute resources used to run the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe the work conforms to it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses potential impacts in the introduction.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing code and assets used in the experiments are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the code in the supplementary materials.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.