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

Jinhao Liang
University of Virginia
jliang@email.virginia.edu

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; however, repeated projection can distort the learned distribution and induce a mismatch with the data manifold. Thus, recent multi-stage procedures attempt to defer projection to "clean" samples during sampling, but they increase algorithmic complexity and accumulate errors across steps. This paper addresses these challenges by proposing a novel *training-free method*, *Chance-constrained Flow Matching* (CCFM), that integrates stochastic optimization into the sampling process, enabling effective enforcement of hard constraints while maintaining high-fidelity sample generation. Importantly, CCFM guarantees feasibility in the same manner as conventional repeated projection, yet, despite operating directly on noisy intermediate samples, it is theoretically equivalent to projecting onto the feasible set defined by clean samples. This yields a sampler that mitigates distributional distortion. Empirical experiments show that CCFM outperforms current state-of-the-art constrained generative models in modeling complex physical systems governed by partial differential equations and molecular docking problems, delivering higher feasibility and fidelity.

1 Introduction

Diffusion and flow-matching generative models have achieved remarkable performance across domains, including image generation [11, 23], robotic motion planning [40, 21], physical system modeling [3, 35], and other scientific applications [39, 29, 5]. 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 or physical laws. For example, physical systems modeling should adhere to partial differential equations, or designing protein-ligand docking must satisfy steric and chemical constraints.

To address these challenges, a growing literature has started exploring constrained handling for generative models. However, these approaches present several notable limitations: *(i)* First, gradient-based guidance methods are dominant in many scientific applications, but, while elegant, such methods cannot guarantee the feasibility of the imposed constraints [12]; *(ii)* Second, repeated projection methods, on the other hand, can enforce strict feasibility, but often at the expense of sample quality [4, 33]; *(iii)* Finally, a more recent line of work attempts to mitigate these issues via an *extrapolation–correction–interpolation* (ECI) procedure [3, 35], which reduces sample distortion. However, the performance of these methods strongly depends on the accuracy of the one-step extrapolation step and constraint formulation, leaving the integration of hard constraints during sampling largely ad hoc and often accompanied by increased overhead, without a unifying theoretical foundation. We provide a detailed discussion of related work in Appendix A.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: ML x OR: Mathematical Foundations and Operational Integration of Machine Learning for Uncertainty-Aware Decision-Making.

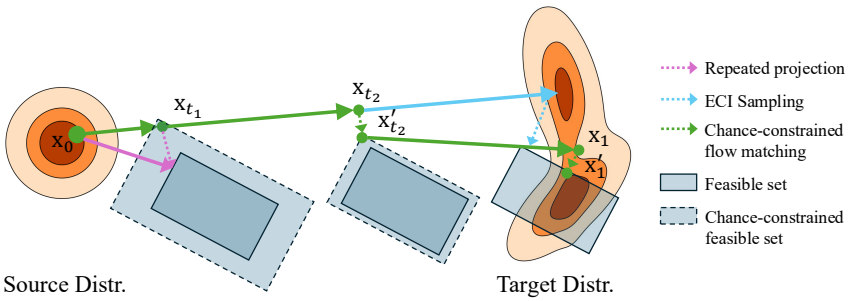


Figure 1: Overview of chance-constrained flow matching (CCFM). Solid arrows follow the sampling path; dashed arrows indicate projection steps. In CCFM, early states remain unchanged (x_{t_1}), intermediate states are minimally corrected ($x_{t_2} \to x'_{t_2}$), and the final state is guaranteed feasible (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.

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. Fidelity here goes beyond feasibility, capturing closeness to the target distribution, such as realistic bound structures in molecular docking or low mean squared error against ground-truth solutions in PDEs. The proposed ideas rely on a key observation: under the standard flow matching parametrization using the optimal transport (OT) path, each intermediate (noisy) state can be expressed in an affine relation to the initial state (noise) and the clean terminal sample [25, 24]. This allows us to make a novel connection between flow matching and chance-constrained programming (CCP) [15], an optimization framework widely used in operations research to enforce constraints with probabilistic guarantees. Modeling the initial state as a random variable and posing the constraint enforcement as a chance-constrained program allows us to derive a feasible set on noisy samples that is theoretically equivalent to the feasible set defined on clean samples (for a chosen risk level α). This enables direct projection of intermediate states, avoiding extrapolation used in ECI sampling, while remaining theoretically equivalent to projection on the clean sample (with probability $1-\alpha$) and thus preserving feasibility. A schematic illustration of the idea is provided in Figure 1.

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 through 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 two scientific domains: molecular docking, specifically modeling the structural transformation between unbound and bound protein conformations, and modeling physical systems governed by PDEs, including Reaction–Diffusion and Navier–Stokes systems. In both domains, CCFM produces samples that satisfy complex constraints and physical principles, achieving new state-of-the-art performance.

2 Methodology

This section proposes CCFM, a novel integration of flow matching with chance-constrained programming to generate high-fidelity samples with feasibility guarantees. A review of the foundational concepts of generative flow matching and chance-constrained programming is provided in Appendix B. We begin by revisiting the classical repeated projection method. Then, we investigate how to incorporate chance constraints into flow matching, followed by an equivalent tractable reformulation that enables efficient implementation. Finally, we provide a theoretical analysis that examines feasibility guarantees and shows equivalence to repeated projection and ECI sampling.

Repeated Projection. Flow matching evolves an initial noise sample toward a target sample via the learned vector field using explicit Euler update (see (6)). To enforce the feasibility of the generated samples, several works apply *repeated projections* [4, 20]. Let $\mathcal{C} = \{x : g(x) \leq 0\}$ denote thes

feasible region. At each step, a projection operator is introduced to map intermediate samples onto $\mathcal{C}$: $\mathcal{P}(x_t) = \arg\min_{x'_t \in \mathcal{C}} \|x'_t - x_t\|_2^2$, where x'_t is the nearest feasible point satisfying the constraints. With this operator, the (projected) sampling rule (6) becomes

$$x_{t+\Delta t} = \mathcal{P}(x_t + \Delta t\, u_\theta(x_t, t)), \tag{1}$$

that is, each tentative Euler step is followed by a projection onto $\mathcal{C}$. This projected-Euler scheme preserves per-step feasibility (when $\mathcal{C}$ is convex).

Observe that the projection operator $\mathcal{P}(\cdot)$ is applied to intermediate noisy samples, while the constraint set is defined with respect to the clean sample. As discussed earlier, this mismatch has been shown to alter the sampling trajectory relative to the learned flow, potentially distorting the induced distribution [18, 32]. To address this challenge, *Chance-constrained Flow Matching* reformulates Problem 1 via a stochastic optimization lens which allows us to adaptively adjust the constraints tightness over the sampling time t.

To do so, we exploit a key property of FM: the linearity of the flow under the OT path, as shown in (7). This linear structure allows us to characterize the dependency between the clean final state x_1, any intermediate state x_t, and the initial noise x_0. In particular, the final state x_1 can be expressed in closed form as a function of x_t and x_0 by rearranging Equation 7 as

$$x_1 = t^{-1}x_t - (1-t)t^{-1}x_0. \tag{2}$$

Since the constraints are defined on the clean sample x_1, we substitute Equation 2 into constraints $g(x_1) \leq 0$, yielding $g\big(t^{-1}x_t - (1-t)t^{-1}x_0\big) \leq 0$. Note that, because x_0 is sampled from the noise distribution, the term $(1-t)t^{-1}x_0$ is random. This allow us to introduce a random variable $\xi = (1-t)t^{-1}x_0$ which isolates the stochastic component, and rewrite the constraint compactly as $g(t^{-1}x_t - \xi) \leq 0$. This reformulation isolates the source of uncertainty explicitly and naturally motivates a chance-constrained treatment at each time step t:

$$\mathbb{P}_\xi\big(g(t^{-1}x_t - \xi) \leq 0\big) \geq 1 - \alpha, \tag{3}$$

where $\xi := (1-t)t^{-1}x_0$ and $\mathbb{P}_\xi$ denotes probability with respect to the distribution of ξ. The corresponding chance-constrained projection operator is then defined as the closest feasible adjustment of x_t that satisfies the probabilistic constraints: $\mathcal{P}_{\text{cc}}(x_t, 1-\alpha) = \arg\min_{x'_t} \|x'_t - x_t\|_2^2$ s.t. $\mathbb{P}_\xi\big(g(t^{-1}x_t - \xi) \leq 0\big) \geq 1 - \alpha$. Then, after embedding the projection operator $\mathcal{P}_{\text{cc}}$ after each sampling step, the update rule for the CCFM sampling process is given by $x_{t+\Delta t} = \mathcal{P}_{\text{cc}}\left(x_t + \Delta t\, u_\theta(x_t, t), \phi(t)\right)$, where $\phi(t) = (t/2)^n$ serves as a probabilistic scheduler that adaptively modulates the satisfaction, where n is a user-specified value. As the sampling time t increases, the contribution of x_0 to x_t decreases, while the probability of satisfaction of the constraints increases simultaneously. These effects yield an adaptive tightening of the constraints, and as $t \to 1$, the random variable ξ vanishes, leading the chance-constrained projection operator to degenerate into its deterministic counterpart. The detailed procedure of CCFM is summarized in Algorithm 1 ((Appendix D)). A detailed theoretical analysis of CCFM, including feasibility guarantees and corresponding proofs, is presented in Appendix C.

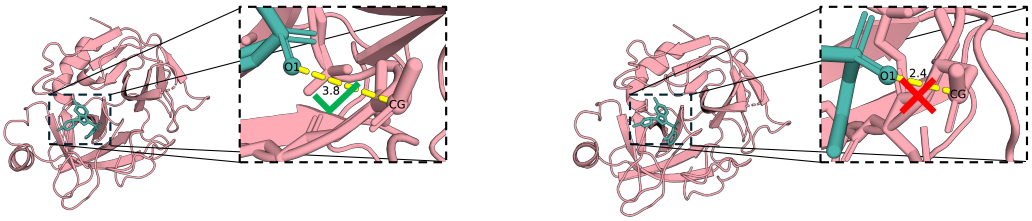
3 Experiments

This section evaluates CCFM against state-of-the-art constrained generative models on two tasks in diverse scientific domains: **Molecular Docking** and **PDE Solution Generation**. We focus on two key aspects of constrained generation: (i) satisfaction of hard constraints (feasibility) and (ii) sample fidelity. Additional experimental results are presented in Appendix F.

Experimental Setting **(1) Molecular Docking:** We use the PDBBind benchmark [27] to predict ligand poses, requiring generated samples to be physically and geometrically plausible. We compare against *FlexDock* [7], the current state-of-the-art framework. **(2) PDE Solution Generation:** We model two complex systems, the 1D *Reaction–Diffusion* equation and the 2D *Navier–Stokes* equation, on a challenging test set of 90 unseen initial conditions. Baselines include the state-of-the-art *PCFM* [35], *DiffusionPDE* [12], and an unconstrained *FFM* [16] backbone. For both tasks, we evaluate *Fidelity* (e.g., RMSD, MMSE) and *Feasibility* (e.g., PoseBusters validity, Constraint Violation). Full details on experimental setting and implementation details are provided in Appendix E.

Table 1: PDBBind docking results 1 generated sample and 2 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.

Methods	**L-RMSD (%)**		**A-RMSD (%)**	**PB Valid (%)**			**Running**
	< 1Å	< 2 Å	< 2 Å	**All**	**+L** < 2Å	**+A** < 2Å	**Time (s)**
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



(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.

Performance Evaluation for Molecular Docking Table 1 illustrates that *CCFM* outperforms the state-of-the-art baseline *FlexDock*, *with particularly large gains*. As shown in Figure 2a, CCFM produces a feasible structure with a distance of 3.8 Å that satisfies the constraint, whereas FlexDock in Figure 2b generates an infeasible structure with a distance of 2.4 Å.

Problem	**Metric**	**CCFM (ours)**	**PCFM**	**DiffusionPDE**	**FFM**
Reaction–Diffusion	MMSE $\times 10^{-2}$ (↓)	**3.3**	3.6	5.2	5.3
	SMSE $\times 10^{-2}$ (↓)	**2.7**	2.9	3.8	3.8
	CV (IC) $\times 10^{-2}$ (↓)	**0**	**0**	8.0	8.2
	CV (CL) $\times 10^{-15}$ (↓)	**9.5**	**9.5**	3.9×10^{10}	3.9×10^{10}
Navier–Stokes	MMSE $\times 10^{-1}$ (↓)	**0.8**	1.7	1.4	1.5
	SMSE $\times 10^{-1}$ (↓)	**0.8**	1.6	8.8	9.0
	CV (IC) (↓)	**0**	**0**	0.12	0.13
	CV (CL) $\times 10^{-15}$ (↓)	3.9	**3.4**	1.2×10^{10}	1.2×10^{9}

Table 2: 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.

Performance Evaluation for PDE We next focus on generating PDE solutions. Table 2 summarizes the performance of the different models tested. For the Reaction–Diffusion, *CCFM* achieves a substantial accuracy gain (MMSE 3.3×10^{-2}, SMSE 2.7×10^{-2}) while providing **zero constraint violations**. On the Navier–Stokes task, the increase in dimension and the setting of the distribution test exacerbate the limitations of the baseline methods. *CCFM* achieves the best overall performance (MMSE 8.4×10^{-2}, SMSE 8.3×10^{-2}), while simultaneously ensuring feasibility.

4 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. This enables the effective enforcement of hard constraints without introducing multi-stage procedures. Our analysis showed equivalence to projecting onto the clean feasible set, thus providing guarantees on constraint satisfaction. The empirical evaluation against the current state-of-the-art on molecular docking and PDE generation showed that CCFM attains strong gains.

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.

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] 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.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

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

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

[12] 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.

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

[14] 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`.

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

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

[17] Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Aleksandr Korotin. Optimal flow matching: Learning straight trajectories in just one step. *Advances in Neural Information Processing Systems*, 37:104180–104204, 2024.

[18] 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.

[19] 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.

[20] 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.

[21] Jinhao Liang, Sven Koenig, and Ferdinando Fioretto. Discrete-guided diffusion for scalable and safe multi-robot motion planning. *arXiv preprint arXiv:2508.20095*, 2025.

[22] 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.

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

[24] 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.

[25] 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.

[26] 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.

[27] 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.

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

[29] 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.

[30] 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.

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

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

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

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

[39] 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.

[40] Fan Zhang and Michael Gienger. Robot manipulation with flow matching. In *CoRL 2024 Workshop on Mastering Robot Manipulation in a World of Abundant Data*, 2024.

[41] 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 Related Work

Recent progress on constrained generative modeling seeks to modify the training or inference stages of these models to account for constraints underlying the data-generating processes. Three predominant strategies can be identified: **(1)** *Penalty Training* introduces constraints violation penalties into the training objectives [30, 19, 1]. While simple, such penalties rarely deliver hard guarantees and can be brittle in out-of-distribution settings. **(2)** *Classifier guidance* incorporates an additional penalty term during sampling [12, 14]. These methods offer much flexibility (as surrogate models can be used in place of constraints), but provide no feasibility guarantees and, additionally, degrade under complex constraint structures (e.g., nonlinear or nonconvex constraints). **(3)** Finally, *Projection-based methods* enforce feasibility by projecting samples onto the feasible manifold [4, 20]. However, these projections are applied at every step of the generative process, and when projections are performed in the early, noisy, states, the process has been shown to substantially distort the training data manifold [6, 33], as the projection disrupts the exactness of sampling dynamics. To mitigate this distortion issue, Cheng et al. [3] and Utkarsh et al. [35] develop ECI-style sampling methods that apply projection only at the final clean sample through a one-step extrapolation step. This, however, comes with an increased sampling complexity cost and, as we will show in our experiments, introduces additional errors from final-state prediction and interpolation.

In contrast, the proposed *CCFM* exploits the optimal transport path in flow matching to enable direct projection at intermediate states, mitigating degradation from projection methods and the complexity of ECI sampling, while ensuring feasibility and high-fidelity generation.

B Preliminaries

Flow matching. Flow matching (FM) is a generative framework that learns a time-dependent 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 via solving the ordinary differential equation (ODE) [24]:

$$\frac{d}{dt}\psi_t(x) = u_t(\psi_t(x)), \quad \text{with initial condition} \quad \psi_0(x) = x. \tag{4}$$

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 [25, 16, 17]. The quadratic-cost OT displacement path is implemented conditionally on data $X_1 \sim q$ by mixing with a simple source $X_0 \sim p$ (e.g., $p = \mathcal{N}(0, I)$) [28, 38, 37]:

$$X_t \;=\; \psi_t(X_0|X_1) \;=\; (1-t)\,X_0 \;+\; t\,X_1, \qquad t \in [0,1]. \tag{5}$$

The training loss is given by: $\mathcal{L}_{\mathrm{CFM}}(\theta) \;=\; \mathbb{E}_{X_0\sim p,\, X_1\sim q,\, t\sim\mathrm{Unif}[0,1]}\, \|u_\theta(X_t,t) - (X_1 - X_0)\|^2$, 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 [13]:

$$x_{t+\Delta t} \;=\; x_t \;+\; \Delta t\, u_\theta(x_t,t). \tag{6}$$

If u_θ matches the ground-truth marginal velocity, the induced flow transports p to q. In deterministic settings, where X_1 is determined by X_0, the trajectories are linear [23, 24]:

$$x_t = (1-t)\,x_0 + t\,x_1. \tag{7}$$

C Theoretical Analysis for CCFM

While the chance-constrained projection operator is conceptually appealing, directly optimizing over a probabilistic constraint is generally intractable. Fortunately, the source distribution used in FM is

typically a simple distribution (e.g., Gaussian); this structure enables a tractable reformulation of the chance-constrained projection operator into an equivalent deterministic problem. Such reformulations are well studied in the optimization communities [31]. The following proposition establishes this tractable reformulation:

Proposition 1 (Tractable reformulations of chance-constrained projection operator)**.** *Let $t \in (0, 1]$ and fix a risk level $\alpha \in (0, 1)$. Consider $a, x_t \in \mathbb{R}^d$, $b \in \mathbb{R}$, and $\xi \sim \mathcal{N}(0, \sigma^2 I_d)$. In particular, when $\xi = (1-t)t^{-1}x_0$ with $x_0 \sim \mathcal{N}(0, I_d)$ (the standard linear interpolation path), one has $\sigma = \frac{1-t}{t}$. Denote the Euclidean norm by $\|\cdot\|_2$, and let $z_{1-\alpha} := \Phi^{-1}(1-\alpha)$ and $z_{1-\alpha/2} := \Phi^{-1}(1-\alpha/2)$, where Φ is the standard normal CDF. Then:*

i. **Linear constraints***: For $g(x) := a^\top x - b$, the chance constraint $\mathbb{P}_\xi\big(g(t^{-1}x_t - \xi) \le 0\big) \ge 1-\alpha$ is equivalent to the deterministic constraint: $a^\top x_t \;\le\; t\,b \;-\; t\,\sigma\,\|a\|_2\, z_{1-\alpha}$.*

ii. **Quadratic constraints***: For $g(x) := (a^\top x)^2 - b$ with $b > 0$, the chance constraint $\mathbb{P}_\xi\big((a^\top(t^{-1}x_t - \xi))^2 \le b\big) \ge 1-\alpha$ is enforced by $\big|a^\top x_t\big| \;\le\; t\Big(\sqrt{b} \;-\; \sigma\,\|a\|_2\, z_{1-\alpha/2}\Big)$, if $\sqrt{b} \;\ge\; \sigma\,\|a\|_2\, z_{1-\alpha/2}$.*

Since $z_{1-\alpha} = \Phi^{-1}(1-\alpha)$ is monotone in α and the variance $\sigma = \frac{1-t}{t}$ decreases as $t \to 1$, the effective constraint bound tightens progressively with the sampling time t.

Proof of Proposition 1. Throughout, let $z_{1-\alpha} := \Phi^{-1}(1-\alpha)$ and $z_{1-\alpha/2} := \Phi^{-1}(1-\alpha/2)$. Write $\sigma_a := \sigma\|a\|_2$ and $\mu := t^{-1}a^\top x_t$. Note that $a^\top\xi \sim \mathcal{N}(0, \sigma_a^2)$.

Linear constraints. Consider $g(x) = a^\top x - b$. The chance constraint

$$\mathbb{P}_\xi\big(a^\top(t^{-1}x_t - \xi) - b \le 0\big) \;\ge\; 1-\alpha$$

is equivalent to

$$\mathbb{P}\big(a^\top\xi \;\ge\; \mu - b\big) \;\ge\; 1-\alpha.$$

Since $a^\top\xi \sim \mathcal{N}(0, \sigma_a^2)$, the quantile characterization gives

$$\mu - b \le -\sigma_a z_{1-\alpha}.$$

and

$$a^\top x_t \;\le\; t\,b \;-\; t\,\sigma_a\, z_{1-\alpha} \;=\; t\,b \;-\; t\,\sigma\,\|a\|_2\, z_{1-\alpha}.$$

Quadratic constraints. Consider $g(x) = (a^\top x)^2 - b$ with $b > 0$. The chance constraint

$$\mathbb{P}_\xi\big((a^\top(t^{-1}x_t - \xi))^2 \le b\big) \;\ge\; 1-\alpha$$

can be written as $\mathbb{P}(|S| \le \sqrt{b}) \ge 1-\alpha$, where $S := a^\top(t^{-1}x_t - \xi) = \mu - a^\top\xi \sim \mathcal{N}(\mu, \sigma_a^2)$. By the union bound (Boole's inequality) [36],

$$\mathbb{P}(|S| \le \sqrt{b}) \;\ge\; 1-\alpha \quad \Longleftarrow \quad \begin{cases} \mathbb{P}(S \le \sqrt{b}) \;\ge\; 1-\frac{\alpha}{2}, \\ \mathbb{P}(S \ge -\sqrt{b}) \;\ge\; 1-\frac{\alpha}{2}. \end{cases}$$

Using the Gaussian quantile characterization, these two one-sided constraints are respectively equivalent to

$$\sqrt{b} \;\ge\; \mu + \sigma_a\, z_{1-\alpha/2} \quad \text{and} \quad -\sqrt{b} \;\le\; \mu - \sigma_a\, z_{1-\alpha/2}.$$

Combining them yields

$$\sqrt{b} \;\ge\; \max\{\mu, -\mu\} + \sigma_a\, z_{1-\alpha/2} \quad \Longleftrightarrow \quad |\mu| \;\le\; \sqrt{b} - \sigma_a\, z_{1-\alpha/2}.$$

Equivalently, in terms of x_t,

$$|a^\top x_t| \;\le\; t\Big(\sqrt{b} - \sigma\,\|a\|_2\, z_{1-\alpha/2}\Big).$$

For non-vacuous feasibility, it is necessary that $\sqrt{b} \;\ge\; \sigma_a\, z_{1-\alpha/2}$. Under these conditions, the original two-sided chance constraint is conservatively enforced with risk split $\alpha/2$ on each tail; the bound is tight when $\mu = 0$ (i.e., $a^\top x_t = 0$), since then

$$\mathbb{P}\big(|S| \le \sqrt{b}\big) = \mathbb{P}\big(|\mathcal{N}(0, \sigma_a^2)| \le \sqrt{b}\big) = 1 - 2\big(1 - \Phi(\tfrac{\sqrt{b}}{\sigma_a})\big),$$

and the condition $\sqrt{b} \geq \sigma_a \, z_{1-\alpha/2}$ is necessary and sufficient for $\mathbb{P}(|S| \leq \sqrt{b}) \geq 1-\alpha$.

Finally, for anisotropic noise $\xi \sim \mathcal{N}(0, \Sigma)$, replace $\sigma\|a\|_2$ by $\sqrt{a^\top \Sigma a}$ throughout; the above derivations remain unchanged since $a^\top \xi \sim \mathcal{N}(0, a^\top \Sigma a)$. □

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 2 (Feasibility Guarantee)**.** *At the sampling time $t = 1$, the chance-constrained projection operator $\mathcal{P}_{\mathrm{cc}}$ from Proposition 1 reduces to the standard Euclidean projection onto the deterministic feasible set $\mathcal{F}_{\mathrm{det}} := \{x' \in \mathbb{R}^d \mid g(x') <= 0\}$. That is,*

$$\mathcal{P}_{\mathrm{cc}}(x_1, 1-\alpha) = \operatorname*{arg\,min}_{x_1' \in \mathcal{F}_{\mathrm{det}}} \|x_1' - x_1\|_2^2.$$

Proof. From (3), we have $\xi = (1-t)t^{-1}x_0$. At $t = 1$, it follows that $\xi = 0$. Hence, for any x',

$$\mathbb{P}_\xi\big(g(t^{-1}x_t - \xi) \leq 0\big) = \mathbb{P}_\xi\big(g(t^{-1}x_t) \leq 0\big) = \begin{cases} 1, & g(t^{-1}x_t) \leq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, at $t = 1$, the chance-constrained feasible set coincides with the deterministic set $\mathcal{F}_{\mathrm{det}} = \{x' : g(t^{-1}x_t) \leq 0\}$. Consequently, $\mathcal{P}_{\mathrm{cc}}(x_1, 1-\alpha)$ reduces to the Euclidean projection onto $\mathcal{F}_{\mathrm{det}}$, which completes the proof. □

Proposition 1 and Corollary 2 indicate that for $t < 1$ the operator provides a probabilistic guarantee, while at $t = 1$ it degenerates to the deterministic projection operator, thereby ensuring feasibility. Next, we demonstrate that feasibility at the final state propagates backward along the linear OT path, and show that the projection operators at intermediate times admit an explicit geometric relation to the projection on final clean samples. All the results below assume $g : \mathbb{R}^d \to \mathbb{R}$ proper, lower semicontinuous, and convex, as recurrent in optimization theory. We define the clean feasible set as $\mathcal{C}_1 := \{x_1 \in \mathbb{R}^d : \, g(x_1) \leq 0\}$, which is assumed to be nonempty. Finally, for $t \in (0, 1]$, we assume that the sampling path is $x_t = (1-t)x_0 + tx_1$, and we use the affine map $M_t(x) := \frac{\left(x - (1-t)x_0\right)}{t}$.

Theorem 3 (Pathwise feasibility propagation)**.** *Under the asssumptions above, the probability $\mathbb{P}_\xi$ w.r.t. the degenerate law concentrated at ξ of satisfying the imposed constraints at time $t \in (0, 1]$ is:*

$$\mathbb{P}_\xi\big(g\big(t^{-1}x_t - \xi\big) \leq 0\big) \;=\; \begin{cases} 1, & \textit{if } g(x_1) \leq 0, \\ 0, & \textit{otherwise,} \end{cases} \tag{8}$$

where $\xi := \frac{1-t}{t}x_0$. In particular, the chance constraint 3 holds if and only if $x_1 \in \mathcal{C}_1$.

Proof. Since $t^{-1}x_t = t^{-1}\big((1-t)x_0 + tx_1\big) = x_1 + \frac{1-t}{t}x_0$ and $\xi = \frac{1-t}{t}x_0$, we have the pathwise identity

$$t^{-1}x_t - \xi \;=\; x_1,$$

which holds for every realization of x_0. Applying g to both sides yields $g(t^{-1}x_t - \xi) = g(x_1)$ almost surely, hence the probability is 1 or 0 depending on whether $g(x_1) \leq 0$. □

Theorem 4 (Geometric structure of the samplewise feasible sets)**.** *For every $t \in (0, 1]$,*

$$\mathcal{C}_t(x_0) = (1-t)x_0 + t\,\mathcal{C}_1,$$

where $\mathcal{C}_t(x_0) := \{x_t : \, g(M_t(x_t)) \leq 0\}$ and $\mathcal{C}_1 := \{x_1 : \, g(x_1) \leq 0\}$. Then, let $\mathbb{P}_t$ and $\mathbb{P}_1$ denote Euclidean projections onto $\mathcal{C}_t(x_0)$ and $\mathcal{C}_1$, respectively, for any $x \in \mathbb{R}^d$,

$$\boxed{\mathbb{P}_t(x) = (1-t)x_0 + t\,\mathbb{P}_1\big(M_t(x)\big),} \tag{9}$$

and the minimizer is unique.

Proof. For any $x_t \in \mathbb{R}^d$,

$$x_t \in \mathcal{C}_t(x_0) \iff g\big(M_t(x_t)\big) \le 0 \iff M_t(x_t) \in \mathcal{C}_1.$$

Thus $x_t \in \mathcal{C}_t(x_0)$ iff there exists $x_1 \in \mathcal{C}_1$ with $x_t = (1-t)x_0 + tx_1$, which yields $\mathcal{C}_t(x_0) = (1-t)x_0 + t\,\mathcal{C}_1$.

For the projection identity, substitute $x_t = (1-t)x_0 + tx_1$ with $x_1 \in \mathcal{C}_1$:

$$\min_{x_t \in \mathcal{C}_t(x_0)} \|x_t - x\|_2^2 = \min_{x_1 \in \mathcal{C}_1} \|(1-t)x_0 + tx_1 - x\|_2^2 = \min_{x_1 \in \mathcal{C}_1} t^2 \|x_1 - M_t(x)\|_2^2.$$

Since $\mathcal{C}_1$ is nonempty, closed, and convex, the Euclidean projection $\mathbb{P}_1(M_t(x))$ exists and is unique. Let $x'^* = \mathbb{P}_1(M_t(x))$. The unique minimizer in the original variable is

$$\mathbb{P}_t(x) = (1-t)x_0 + tx'^* = (1-t)x_0 + t\,\mathbb{P}_1\big(M_t(x)\big),$$

as claimed. □

The above is a key result for this work, as it offers a key geometric insight. It implies that projecting onto the time-varying feasible set $\mathcal{C}_t(x_0)$, for any initial state x_0, can be geometrically decomposed: it is equivalent to first mapping the point to the final state via M_t, then performing a single projection onto $\mathcal{C}_1$, and finally mapping the result forward. *Crucially, this shows that the geometry of the feasible space is preserved throughout the generation process.*

Remark 1. Notice that, while it is customary in optimization to assume convexity in the constraint spaces to prove theoretical results, empirically, our results will show that these guarantees hold up in highly non-convex cases. Appendix E details the chance-constraint implementation for both convex and non-convex constraints adopted in our experiments.

Remark 2. In practice, numerical discretization may introduce slight curvature around the ideal straight-line characteristics in (7). Nevertheless, recent theoretical advances show that the ground-truth velocity field can be approximated using Lipschitz neural networks, with approximation error bounded by $(\sqrt{d}+1)\epsilon$ for arbitrary $\epsilon > 0$, where d is the dimension of data [41]. This guarantee ensures that the discrepancy between the learned and ground-truth velocity fields remains small. Moreover, although an approximation error exists, Gao et al. [9] show that both estimation and discretization errors are controllable: the velocity estimation error decays at a rate $\tilde{O}((nt^2)^{-1/(d+3)})$, where n is the number of training samples and $\tilde{O}$ hides polylogarithmic factors. The discretization error scales as $O(\sqrt{\Upsilon})$, thus smaller step sizes Υ directly reduce error at a square-root rate. The use of straight-line characteristics has also been adopted in prior works with strong empirical performance [25, 26], and our experiments further corroborate their effectiveness.

D Detailed Algorithm

Algorithm 1: Chance-constrained Flow Matching

Input: Learned vector field u_θ, Euler steps N.
Sample noise function $x_0 \sim p$.;
for $t \leftarrow 0, 1/N, 2/N, \ldots, (N-1)/N$ **do**
  $x_{t+1/N} \leftarrow x_t + u_\theta(x_t, t)/N$;
  $x_{t+1/N} \leftarrow \mathcal{P}_{\text{cc}}(x_{t+1/N}, \phi(t))$;
return x_1;

Remark 3. CCFM can be contrasted with existing projection strategies. *Repeated projection* enforces the same constraints at every step. When the constraints are too tight, especially early in sampling, the projection may significantly distort the learned velocity. This deviation from the learned distribution reduces generation quality. On the other hand, at each step t, *ECI* predicts the final state in one step, projects it, and interpolates back to the intermediate state. Although it avoids perturbing noisy samples, the velocity depends on a projected one-step prediction rather than the learned distribution, making performance sensitive to prediction quality. In contrast, *CCFM* adaptively adjusts constraint tightness over time t. This yields minimal modifications of the learned velocity toward feasibility and enables high-fidelity generation without reliance on *one-step predictions*.

E Implementation Details

E.1 Experimental setting

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 [27], which provides a standard testbed for evaluating protein–ligand docking methods. Comparisons are made against *FlexDock* [7], 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 also 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) [2] and its conjunction with RMSD thresholds, measuring the physical plausibility of predicted complexes.

PDE Solution Generation. Next, following [35], 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. Unlike [3, 12], which evaluate on only 10 test initial conditions, we adopt 90 unseen (diverse) conditions to yield a challenging and discriminative benchmark.

Comparisons are conducted against several representative baselines: (a) *Physics-Constrained Flow Matching (PCFM)*[35], which is the current state of the art; it extends ECI sampling [3] to nonlinear constraints and reduces to ECI in the linear setting; (b) *DiffusionPDE* [12], which enforces physical priors through penalty terms during iterative sampling; and (c) *Functional Flow Matching (FFM)* [16], a pretrained unconstrained FM. All methods employ the same pretrained FFM backbone and are evaluated with identical sampling steps, differing only in how constraints are embedded into the sampling process.

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.

E.2 Molecular Docking Formulation

Given a receptor with atom coordinates $\mathcal{Y} \in \mathbb{R}^{N_{\mathrm{rec}} \times 3}$ and a ligand template conformer $\mathcal{X}_0 \in \mathbb{R}^{N_{\mathrm{lig}} \times 3}$, 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

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

where $T_{\mathrm{lig}}(\cdot; \phi)$ 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 $(i, j) \in \mathcal{E}_{\mathrm{bond}}$,

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

where $\mathcal{E}_{\mathrm{bond}}$ is the set of ligand covalent bonds, $\delta_b = 0.25$ denotes the tolerance, $\ell_{ij}^{\mathrm{low}}, \ell_{ij}^{\mathrm{up}}$ 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 $(i{-}j{-}k) \in \mathcal{E}_\angle$,

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

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:

E.3 PDE Problem Formulation

E.3.1 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.

E.3.2 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.

E.4 Constraints Enforcement

E.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.

E.4.2 PDE

Following the enforcement strategy of [35], 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 $g_i(x) \leq 0$, 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:

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

Under smoothness and full-rank assumptions, these iterations converge quadratically [35]. In both PCFM and CCFM, we allocate a fixed budget of 30 iterations after the final sampling step to ensure strict feasibility.

E.5 Dataset Details

E.5.1 Molecular Docking

This work utilizes the same dataset as adopted in [7], namely the PDBBind benchmark [27], which has become a standard for evaluating docking models. Following previous work [34, 7], 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 [8] 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 [22]. 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.

E.5.2 PDE

Following the dataset construction strategies in [3, 35], 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 [10], 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. For the test set, a separate pool of data is first generated by combining 90 new initial conditions with 90 new boundary flux configurations; from this pool of 8,100 trajectories, 1,000 are then randomly selected for evaluation. 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. For evaluation, a separate dataset is first generated using 90 new initial conditions and 90 new forcing phases. From this pool of 8,100 trajectories, we then randomly select 1,000 trajectories to form the final test set. The initial conditions used for training and evaluation are disjoint.

E.6 Model Details

E.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) [7]. 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.

E.7 PDE

For all PDE benchmarks, we adopt the Functional Flow Matching (FFM) framework parameterized by a Fourier Neural Operator (FNO) backbone [16]. 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 3 and Table 4 summarize the hyperparameters used during training for reaction–Diffusion and Navier–Stokes equations, respectively.

Table 3: 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 4: 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

E.8 Sampling Setup

E.8.1 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 3 and the remaining results in the appendix F.

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 E.4, followed by a final refinement of 20 iterations once the sampling trajectory is complete.

E.8.2 PDE

We adopt high-order ODE solvers (Dopri5) for unconstrained generation. For all constraint-aware methods, we employ the modified Euler scheme (Heun's method) 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 1, 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 $\phi(t) = (t/2)^n$. The parameter n thus controls the tightening schedule: smaller values (e.g., $n = 0.1$) enforce stricter constraints at later stages, while larger values (e.g., $n = 0.5$) 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 F. 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 [12] 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.

E.9 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.

F Additional Results

F.1 Molecular Docking

Table 5 reports docking results on the PDBBind docking benchmark under varying numbers of generated samples and inference steps. Notice how, across all configurations, *CCFM* consistently outperforms the state-of-the-art baseline *FlexDock*, *with particularly large gains when the computational budget is limited*. Remarkably, with only one sample and two inference steps, CCFM

Table 5: 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	$<$ 2 Å	$<$ 2 Å	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
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
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
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
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
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

achieves comparable L-RMSD and A-RMSD values, confirming that high-fidelity pose generation is preserved. At the same time, CCFM raises PB Valid from 20.34% (FlexDock) to 47.74%, representing a **2.3**$\times$ improvement over FlexDock. 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**$\times$ gain in runtime (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, again matching the performance of FlexDock at 20 samples while using far fewer samples and computational time. Increasing the 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 F.1.

A comparison between CCFM and FlexDock across varying sample sizes and inference steps is shown in Figure 3.

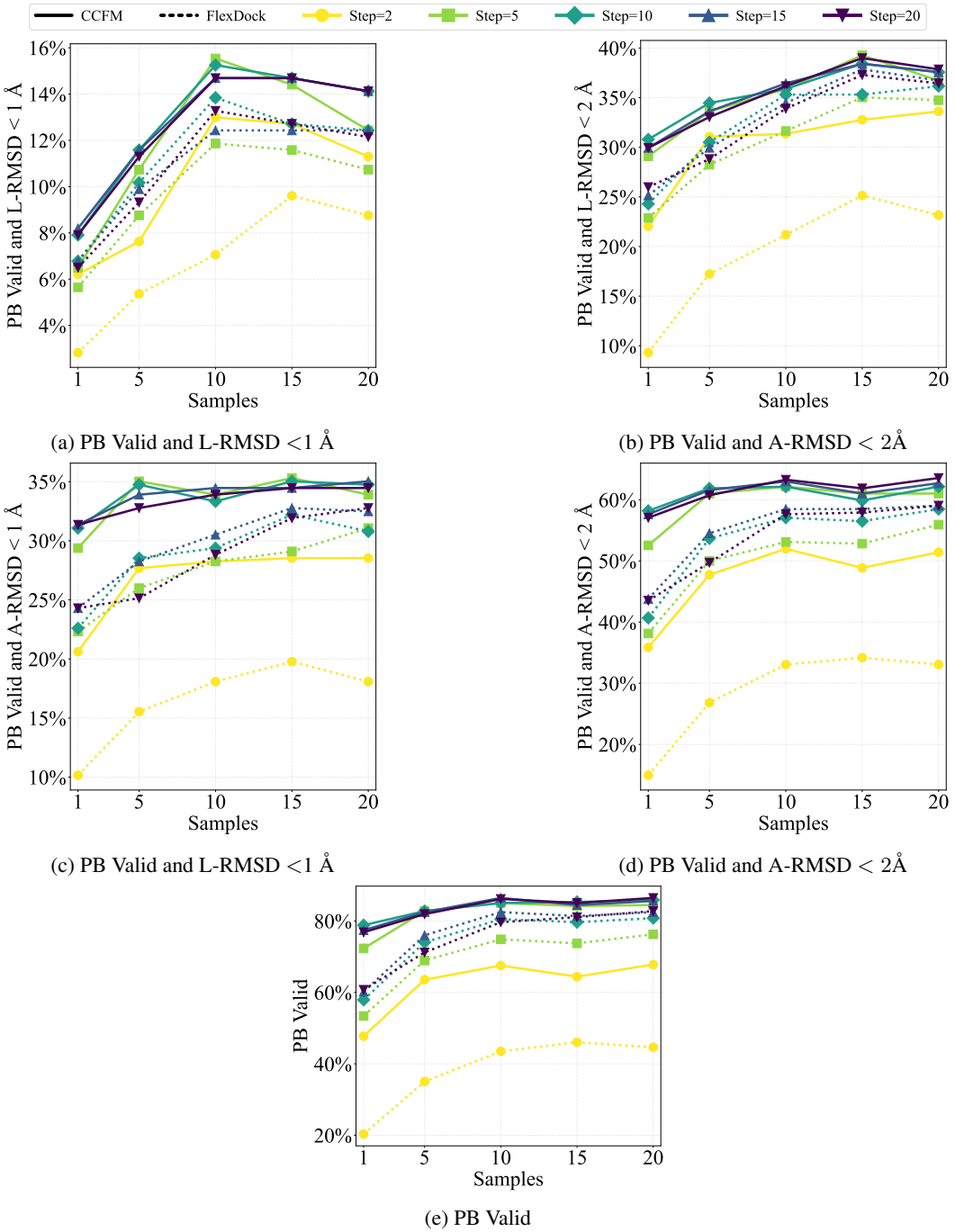
At the strictest setting (PB Valid with L-RMSD $<$ 1 Å, step = 2, sample = 1), CCFM attains over 6% success rate, more than doubling FlexDock's 3%. A similar advantage persists when relaxing the cutoff to L-RMSD $<$ 2 Å, where CCFM consistently surpasses FlexDock across all sample and step budgets.

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.

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



(a) PB Valid and L-RMSD <1 Å

(b) PB Valid and A-RMSD < 2Å

(c) PB Valid and L-RMSD <1 Å

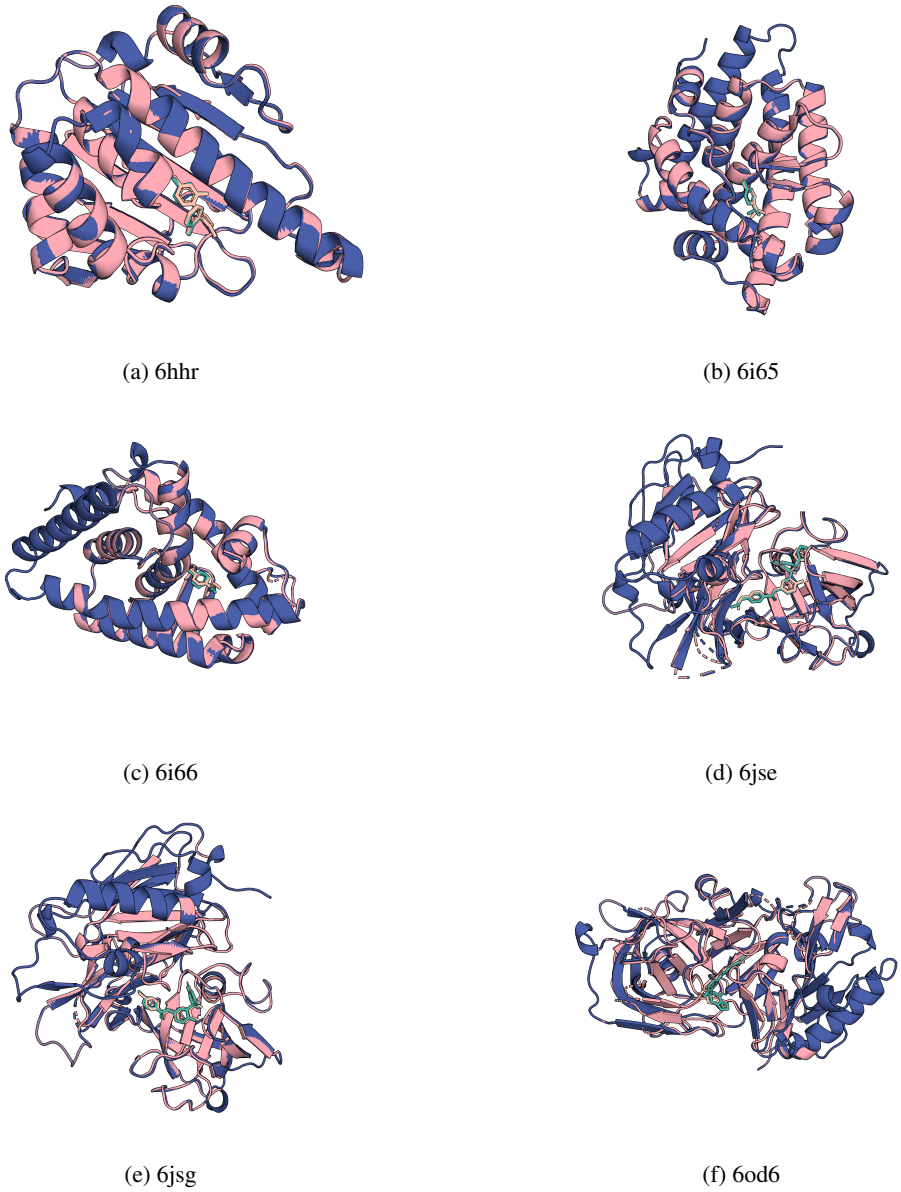
(d) PB Valid and A-RMSD < 2Å

(e) PB Valid

Figure 3: Comparison of CCFM (solid lines) and FlexDock (dashed lines) under different sampling budgets and evaluation metrics.

F.2 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 (Navir–Stokes), the point-wise average residuals varying with physical time, and the comparison of models using an example of the generated solution field.



(a) 6hhr (b) 6i65

(c) 6i66 (d) 6jse

(e) 6jsg (f) 6od6

Figure 4: 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.

Table 6 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 avoids unnecessary projections and reduces computational cost.

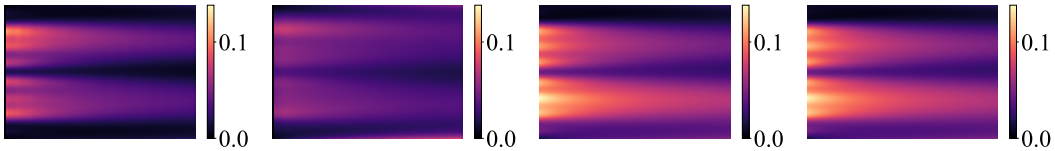
F.2.1 Reaction–Diffusion

Figure 5 illustrates MMSE across physical time, where the prevalence of darker regions indicates improved temporal stability.

Table 7 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

Table 6: 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

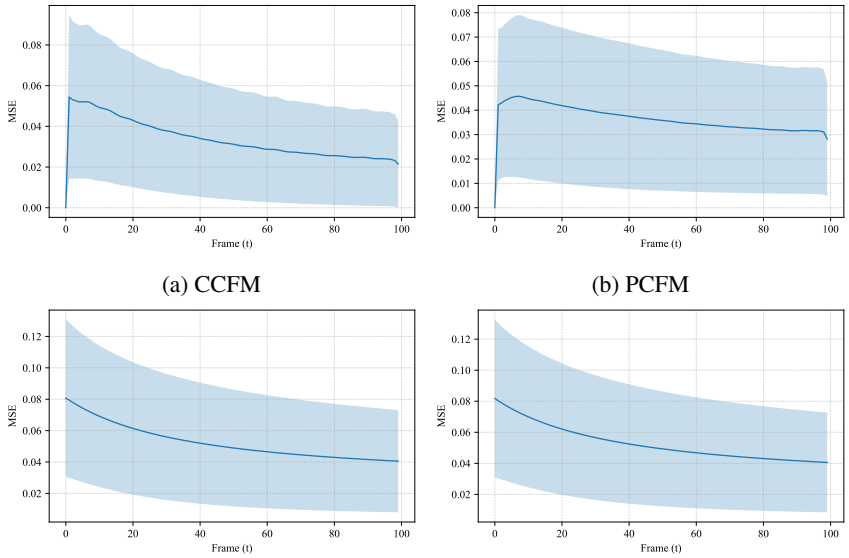


(a) CCFM (ours) (b) PCFM (c) DPDE (d) FFM

Figure 5: MMSE (the darker the better) 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.

fluctuations that remain within a reasonable range. This indicates that the method is robust to the choice of the probability scheduler.

Figure 6 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 7 shows the SMSE, where the x-axis represents physical time and the y-axis represents space. Similar to the MMSE (Figure 5), the proposed CCFM achieves the lowest errors compared with other models. The solution residuals, shown in Figure 8 demonstrates the improvement from CCFM compared with other models in generating a specific solution field. The CCFM-generated sample 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.



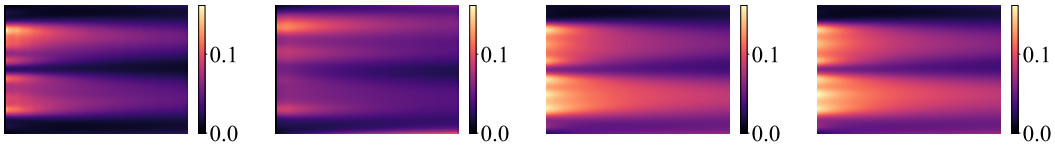
(a) CCFM (b) PCFM

(c) DPDE (d) FFM

Figure 6: Comparison of generated solutions for the Reaction–Diffusion equation in terms of MMSE varying with physical time.

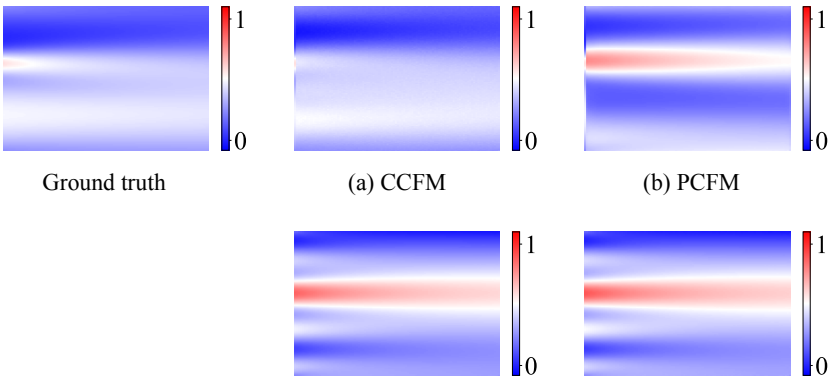
Table 7: Sensitivity analysis for probability of constraint satisfaction. The probability is determined by the function $\phi(t) = (t/2)^n$, and we adjust n to observe the results.

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: 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.



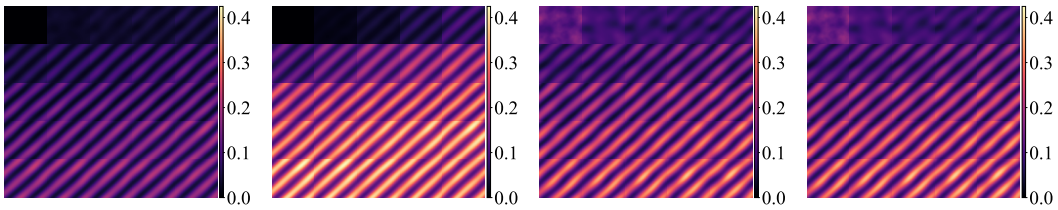
Ground truth (a) CCFM (b) PCFM

(c) DPDE (d) FFM

Figure 8: Example solution fields of the Reaction–Diffusion equation generated using the models compared with the ground truth.

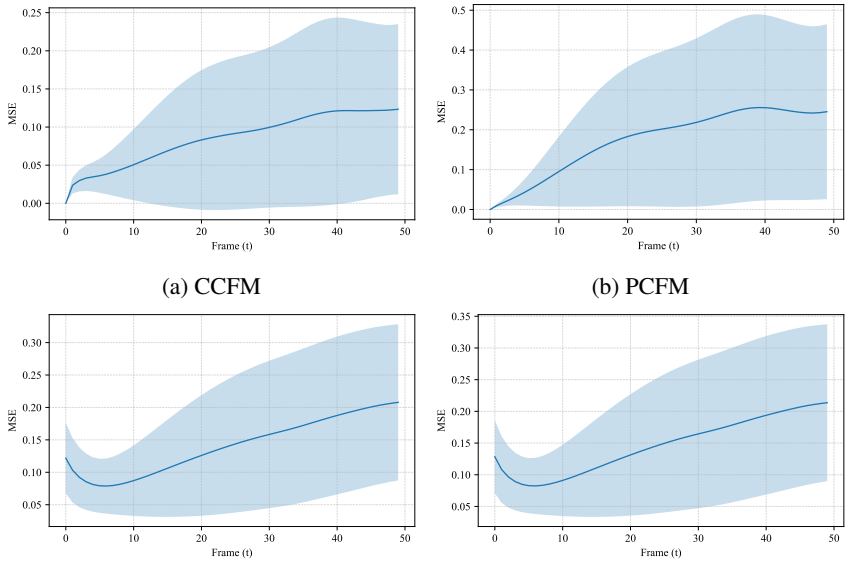
F.2.2 Navier–Stokes

For the Navier–Stokes problem, to complement the results presented in Section 3, we report evaluation results for all physical time frames. Figure 9 visualizes the results, showing that *CCFM* consistently suppresses error accumulation and aliasing. 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 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).



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

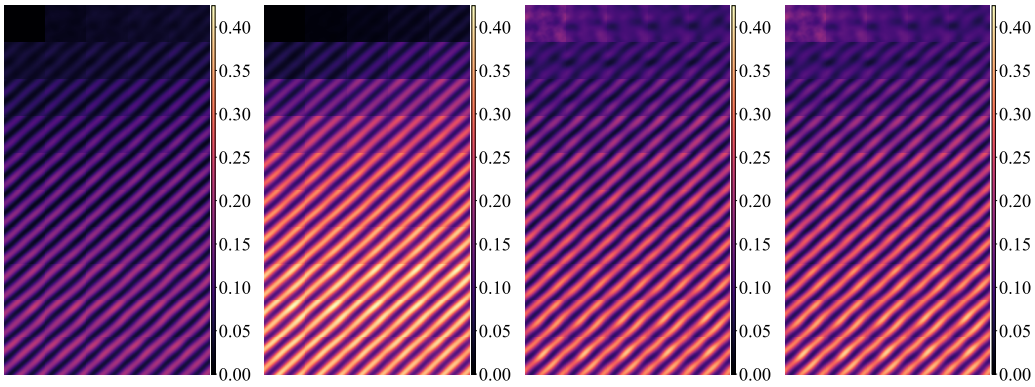
Figure 9: MMSE (the darker the better) of Navier–Stokes solutions over physical time, downsampled from 50 to 25 frames for clarity. Each frame corresponds to the system state at a given physical time.



(a) CCFM (b) PCFM

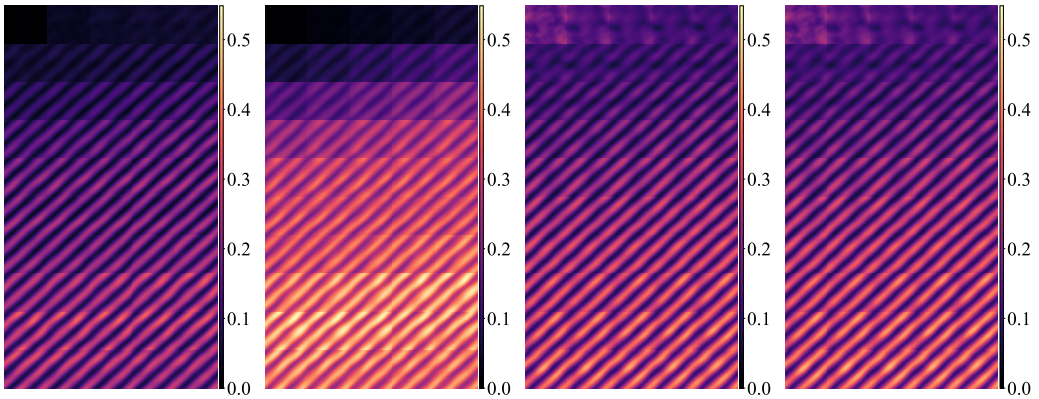
(c) DPDE (d) FFM

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



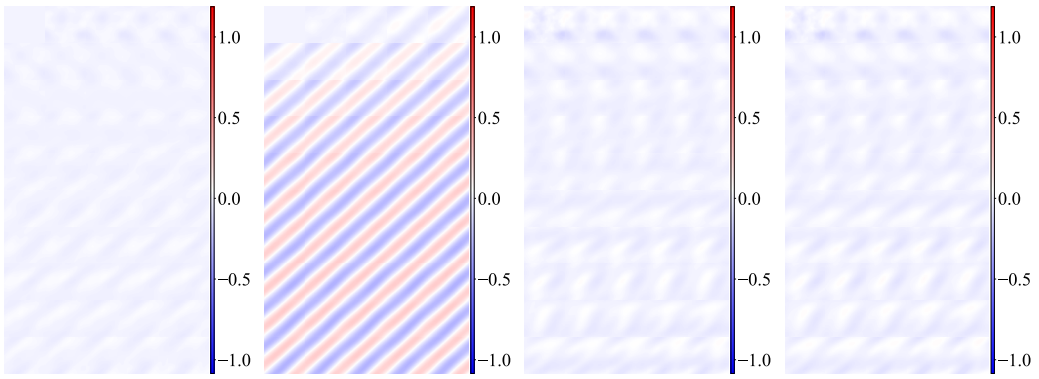
(a) CCFM (b) PCFM (c) DPDE (d) FFM

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.



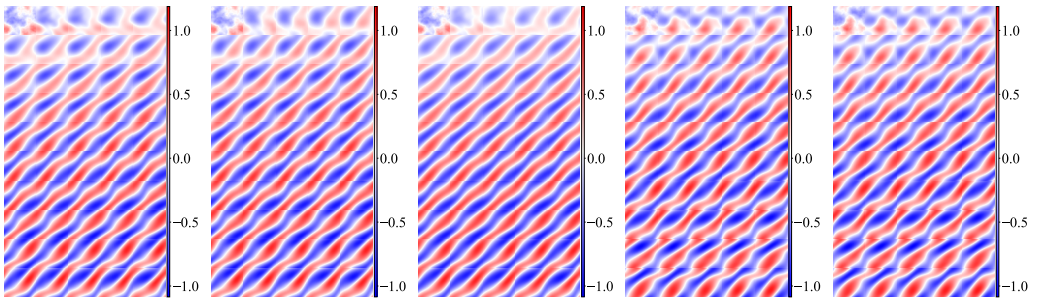
(a) CCFM (b) PCFM (c) DPDE (d) FFM

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.