PocketVE: Stable and Property-Guided Structure-Based Drug Design with Variance-Exploding Diffusion

Peining Zhang
University of Connecticut
Storrs, CT 06269, USA
peining.zhang@uconn.edu

Jinbo Bi
University of Connecticut
Storrs, CT 06269, USA
jinbo.bi@uconn.edu

Abstract

Protein-conditioned 3D molecule generation is a central challenge in structure-based drug design, requiring a balance between pocket compatibility, molecular properties, and physical geometry. We propose **PocketVE**, a protein-pocket-conditioned variance-exploding (VE) diffusion framework that couples stable coordinate denoising with inference-time property guidance. Specifically, PocketVE combines an EDM-style training and sampling setup for 3D denoising, classifier-free guidance for multi-property steering, and adaptive protein perturbation as a training-time pocket regularizer. On the CrossDocked2020 benchmark under GenBench3D, PocketVE improves Valid_{3D} from 58.6 to 80.6 and reduces strain energy from 457.4 to 127.9 relative to its guided TAGMol architectural parent; relative to TargetDiff, it attains comparable Valid_{3D} with lower strain energy (127.9 vs. 306.0), while retaining competitive docking and molecular-property scores under moderate guidance. A guidance-scale study shows that moderate guidance gives a favorable balance between target-related objectives and geometric quality, whereas stronger guidance can degrade geometry and distributional fidelity. Pocket-permutation and PoseCheck diagnostics further support pocket-specific spatial compatibility with reduced steric conflicts. Overall, the results suggest that geometric stability and inference-time property guidance should be considered as coupled design objectives.

1 Introduction

Generating drug-like molecules with desirable properties and high affinity to a given protein binding site, a task known as structure-based drug design (SBDD), sits at the intersection of 3D geometry and molecular property optimization. A generated ligand must satisfy two classes of constraints: pocket-dependent ones (shape complementarity, binding pose, steric fit) and intrinsic molecular ones (drug-likeness, synthetic accessibility, logP) [23, 45]. Generative models for molecular design have progressively transitioned from 1D strings [11, 28] and 2D graphs [25, 39] to direct 3D modeling [31, 34]. Among 3D generative approaches, non-autoregressive diffusion models have substantially advanced target-aware generation [17, 40, 19]: TargetDiff [14] introduced SE(3)-equivariant diffusion to jointly denoise the full ligand without relying on a fixed generation order, while TAGMol [7] demonstrated that gradient-guided sampling can steer the generative distribution toward regions of higher binding affinity.

Despite these advances, integrating strong protein-ligand interactions without deteriorating molecular geometry remains an open challenge. First, the reverse sampling trajectory can be sensitive to the conditioning signal, creating a trade-off between steerability and stability. While image models often work with bounded pixels or latent representations [37], where strong guidance can lead to overexposed outputs [47, 30], guidance in SBDD acts directly on unconstrained Euclidean coordi-

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

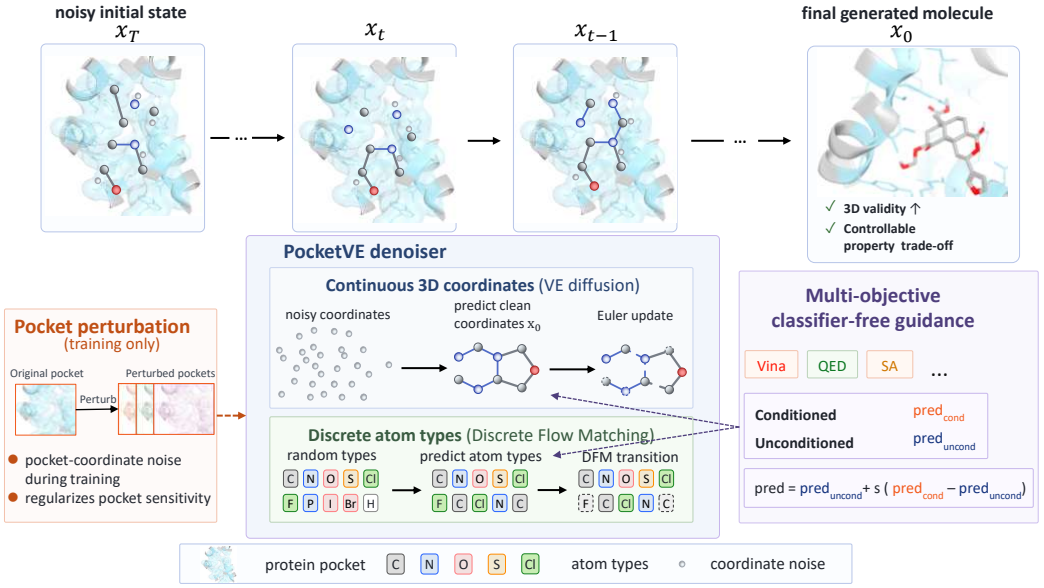


Figure 1: Overview of PocketVE. PocketVE couples VE-based coordinate denoising with discrete flow matching for atom types. Multi-objective classifier-free guidance provides inference-time property steering, while protein-pocket perturbation regularizes the denoiser during training.

nates. Consequently, excessive guidance pressure often drives the geometry toward non-physical conformations, as reflected by the atom clashes and high strain energy reported in recent GenBench3D evaluations [1], where higher strain indicates a less physically plausible 3D conformation. Second, existing models primarily focus on learning the unconditional chemical distribution of the training data, leaving the integration of real-world multi-objective properties highly inflexible and poorly transferable [7, 24, 6]. As a result, adjusting how strongly a trained model pursues the protein target or a specific drug-like profile usually requires training a separate time-dependent classifier for every property of interest, making it poorly scalable when balancing multiple objectives.

Motivated by these challenges, this paper studies how conditional control changes the property–geometry trade-off in protein-conditioned 3D generation. Our main point is that the guidance mechanism and coordinate backbone should be designed jointly, because both shape the reverse trajectory in Euclidean space. In particular, a VE-style coordinate backbone provides a natural basis for stable denoising across noise scales, while pocket-aware scaling and sampling choices determine how that backbone interacts with the protein coordinate frame. We therefore evaluate PocketVE as an integrated target-aware framework, without attributing its overall improvement to VE alone. PocketVE combines VE/EDM-style coordinate parameterization, pocket-aware scale preservation, classifier-free guidance for multi-property steering, and training-time pocket perturbation. Figure 1 summarizes the overall pipeline and the three design axes of PocketVE.

We evaluate our framework on the CrossDocked2020 benchmark under the GenBench3D protocol [1]. Relative to guided TAGMol, PocketVE increases Valid_{3D} from 58.6 to 80.6 and reduces strain energy from 457.4 to 127.9, while retaining competitive docking and molecular-property scores under moderate guidance. TargetDiff and PAFlow provide complementary geometry and affinity references: PocketVE is comparable to TargetDiff in Valid_{3D} and has lower strain, whereas PAFlow obtains stronger Vina scores at a substantial geometric cost. Guidance experiments further show a trade-off between target-related properties and geometric fidelity. Together, these findings suggest that geometric stability and conditional steering should be considered jointly in target-aware 3D molecular generation.

Our main contributions are as follows:

- We develop a pocket-aware VE coordinate parameterization [27] that preserves the shared protein–ligand coordinate frame while using EDM-style preconditioning for stable denoising, together with discrete flow matching for atom types.

- We characterize the guidance-scale trade-off in protein-conditioned 3D molecule generation, showing that Vina Score, QED, and SA respond differently from geometry metrics such as Valid_{3D} and strain energy.
- We combine this backbone with multi-objective CFG and protein-side perturbation, improving the balance between structural validity, drug-like properties, and target-specific affinity on CrossDocked2020 under the GenBench3D protocol.

2 Related Work

Target-Agnostic 3D Molecular Generation. In target-agnostic 3D molecular generation, Equivariant Diffusion Models learn coordinate denoising with E(3)-equivariant networks [19]. Subsequent works extend this paradigm to latent representations and joint structure–geometry modeling [44, 43], as well as unified generation of atom types, coordinates, and bonds via flow matching [22]. More recently, VE-style parameterizations further improve geometric stability in 3D generation [46]. These advances suggest that geometric stability should be built into the generative backbone rather than enforced post hoc. PocketVE brings this design principle into the protein-conditioned setting, where ligand coordinates must remain consistent with an explicit pocket geometry.

Structure-Based Drug Design. Early deep generative models for structure-based drug design (SBDD), such as 3D-SBDD [31] and Pocket2Mol [34], established spatial representations and architectures for pocket-conditioned generation. However, many of these methods rely on autoregressive atom or motif placement, which is prone to error accumulation along a fixed generation order. Non-autoregressive diffusion methods instead denoise all ligand atoms jointly. TargetDiff [14] introduced an SE(3)-equivariant formulation for target-aware 3D diffusion, while DiffSBDD [38] further developed equivariant diffusion models for structure-based generation. IPDiff [21] incorporates protein–ligand interaction priors, BindDM adaptively extracts interaction-relevant protein–ligand subcomplexes [20], PAFlow [49] adopts prior-guided flow matching, and PocketXMol studies direct clean-coordinate prediction in pocket-conditioned molecular design [33]. Flow-based formulations are attractive for sampling efficiency, but recent evidence suggests that low-step generation alone does not guarantee chemically reliable 3D geometry [32]. PoseCheck [16] analyzes generated protein–ligand poses directly and shows that physical violations or missing key interactions may be obscured when evaluation relies on redocking. GenBench3D [1] further emphasizes the 3D conformation quality of generated ligands. For example, it evaluates whether generated bond lengths and valence angles are consistent with reference distributions from 3D structure databases such as CSD [12] and LigBoundConf [42]. Together, these findings motivate our focus on the property–geometry trade-off: Vina-based docking scores and molecular properties should be interpreted together with pose and conformation quality.

Conditional Diffusion. Mainstream strategies for injecting conditional signals modify the reverse sampling trajectory via classifier guidance [6] or classifier-free guidance (CFG) [18]. However, applying these paradigms to 3D molecular design is non-trivial: stronger conditional pressure can degrade physical geometry, causing atom clashes or pushing molecules off-distribution [1]. TAGMol [7] uses external classifier guidance to steer sampling toward desired molecular properties, but this requires separate property predictors and scale calibration. BADGER [24] further studies guidance in diffusion-based SBDD, combining classifier guidance and classifier-free guidance for binding-affinity control and extending the same framework to joint optimization over affinity, QED, and SA. Other approaches, including alignment methods such as AliDiff [13] and reinforcement-learning-based guidance [50], directly optimize or fine-tune the generator toward preferred properties. In particular, we study this trade-off under a VE-style 3D backbone, where stronger conditional control may improve target objectives while introducing geometric drift.

3 Method

3.1 Overview

We model the denoising process over a hybrid state space that contains continuous coordinates and discrete atom types. The task of structure-based drug design (SBDD) is to learn a conditional

distribution $p(\mathcal{L} \mid \mathcal{P})$ over ligands conditioned on a protein pocket. A ligand with N_L atoms is represented as $\mathcal{L} = (\mathbf{x}, \mathbf{z})$, where $\mathbf{x} \in \mathbb{R}^{N_L \times 3}$ are atom coordinates and $\mathbf{z} \in \{0,1\}^{N_L \times S}$ are atom-type indicators. The protein pocket is written as $\mathcal{P} = (\mathbf{y}, \mathbf{c})$, where $\mathbf{y} \in \mathbb{R}^{M \times 3}$ are pocket atom coordinates and $\mathbf{c}$ are pocket atom features. Our goal is to model $p_\theta(\mathcal{L} \mid \mathcal{P})$, so that the generated ligand is both geometrically valid and compatible with the target pocket.

Our framework comprises three coupled components. It uses an EDM-style [27] generation backbone to stabilize continuous coordinate denoising, classifier-free guidance to control auxiliary property targets at inference time, and training-time protein perturbation to reduce over-reliance on a single rigid pocket realization.

3.2 EDM-style generation backbone

For the continuous coordinates, we use a variance-exploding (VE) forward process. Given a clean ligand coordinate set $\mathbf{x}_0$, we sample a noise level $\sigma \sim p(\sigma)$ and form

$$\mathbf{x}_\sigma = \mathbf{x}_0 + \sigma\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{1}$$

In practice, the noise scale σ is sampled from a log-normal distribution: $\sigma \sim \mathrm{LogNormal}\Big(\ln\sqrt{\sigma_{\min}\sigma_{\max}}, \big[\frac{1}{8}\ln(\sigma_{\max}/\sigma_{\min})\big]^2\Big)$, so σ lies near the range $[\sigma_{\min}, \sigma_{\max}]$ in most cases and follows the usual VE/EDM-style noise schedule [27].

For atom types, we use a separate discrete corruption process. Following the discrete flow matching view [4], this branch learns a categorical denoising target over the same hybrid state space. We use the corruption kernel

$$q(\mathbf{z}_\sigma \mid \mathbf{z}_0) = (1 - m(\sigma))\,\delta(\mathbf{z}_\sigma = \mathbf{z}_0) + m(\sigma)\,\mathrm{Unif}(K), \tag{2}$$

where $\mathrm{Unif}(K)$ is the uniform distribution over atom types. The masking schedule is $m(\sigma) = \frac{\ln\sigma - \ln\sigma_{\min}}{\ln\sigma_{\max} - \ln\sigma_{\min}}$. This keeps the continuous coordinate diffusion and the discrete type diffusion aligned at the same noise level.

A small but important target-aware detail is the input scaling used before the equivariant backbone. In standard EDM, the noisy sample is often scaled as $\mathbf{x}_\sigma/\sqrt{\sigma^2 + \sigma_{\mathrm{data}}^2}$ before entering the backbone. In our setting, however, ligand coordinates interact with unscaled protein-pocket coordinates inside the same distance-based geometric graph. Directly using the standard factor would shrink low-noise ligand coordinates by $1/\sigma_{\mathrm{data}}$, while leaving protein coordinates unchanged, thereby distorting protein–ligand distances. We therefore use the scale-preserving input $\tilde{\mathbf{x}}_\sigma = \frac{\sigma_{\mathrm{data}}}{\sqrt{\sigma^2+\sigma_{\mathrm{data}}^2}}\mathbf{x}_\sigma$, which recovers the original ligand scale as $\sigma \to 0$ and still attenuates high-noise inputs. This preserves the physical coordinate frame shared by ligand and pocket atoms. We use $\sigma_{\mathrm{data}} = 10.0$ as a fixed coordinate-scale parameter, rather than estimating it as a per-ligand sample variance or normalizing each ligand independently. This value covers both the internal ligand extent and the ligand displacement from the pocket center under pocket-centered normalization.

For the coordinate branch, we then use the EDM preconditioning form

$$D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}) = c_{\mathrm{skip}}(\sigma)\mathbf{x}_\sigma + c_{\mathrm{out}}(\sigma)\,F_\theta(\tilde{\mathbf{x}}_\sigma;\, c_{\mathrm{noise}}(\sigma),\, \mathbf{z}_\sigma,\, \mathcal{P})\,. \tag{3}$$

Here F_θ is the equivariant backbone shared by both branches, and the skip term keeps the coordinate prediction anchored to the noisy input. This plays the same role as the preconditioning in VEDA [46], but here it is conditioned on the pocket. Further coefficient details are provided in Appendix B.

Atom types are recovered through a categorical prediction module built on top of the same equivariant features. Its output $H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}) \in \mathbb{R}^{N_L \times K}$ gives the logits for the clean atom types. The clean ligand is then recovered by

$$\hat{\mathbf{x}}_0 = D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}), \qquad \hat{\mathbf{z}}_0 = \mathrm{softmax}\big(H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P})\big). \tag{4}$$

The training objective is a joint denoising loss over geometry and atom identity:

$$\mathcal{L}_{\mathrm{base}}(\theta) = \mathbb{E}_{\mathcal{L}_0, \mathcal{P}, \sigma}\Big[\lambda_x(\sigma)\,\|\hat{\mathbf{x}}_0 - \mathbf{x}_0\|_2^2 + \lambda_z(\sigma)\,\mathrm{CE}(\hat{\mathbf{z}}_0, \mathbf{z}_0)\Big]. \tag{5}$$

3.3 Sampling procedure

At inference time, we follow a fixed decreasing VE noise schedule. At each step, the coordinate branch predicts the clean structure and applies a first-order Euler update:

$$\mathbf{d}_i = \frac{\mathbf{x}_{\sigma_i} - D_\theta(\mathbf{x}_{\sigma_i}, \mathbf{z}_{\sigma_i}, \sigma_i, \mathcal{P})}{\sigma_i}, \qquad \mathbf{x}_{\sigma_{i+1}} = \mathbf{x}_{\sigma_i} + (\sigma_{i+1} - \sigma_i)\mathbf{d}_i. \tag{6}$$

Ligand coordinates are initialized around the pocket center, and ligand atom types are initialized from the uniform discrete prior. For the discrete atom types, we update $\mathbf{z}_{\sigma_i}$ with a DFM-based discrete sampler driven by the categorical predictor H_θ. The number of ligand atoms is sampled from a prior conditioned on the estimated pocket size, matching the inference-time size prior used in TAGMol [7]. When classifier-free guidance is enabled, we use the same sampler with a tunable guidance weight. The concrete noise range, generalized arcsin schedule, discrete-sampler details, and sampling-step ablations are provided in Appendix C and Appendix D.

3.4 Classifier-free guidance

We focus on three commonly used generation-quality objectives: Vina for predicted binding affinity, QED for drug-likeness, and normalized SA for synthetic accessibility. To make generation steerable, we discretize each target property into percentile bins computed on the training set and encode the resulting bin indices as an auxiliary condition $\mathbf{a}$. Multi-objective control at inference time is specified by choosing one target bin per property and concatenating the resulting indices into the same auxiliary condition vector. We apply condition dropout [18] to this auxiliary input during training while always keeping the pocket condition $\mathcal{P}$. For notational simplicity, this auxiliary input was suppressed in the backbone definition above and is written explicitly only in this subsection. Let $\varnothing$ denote the dropped auxiliary condition. Following the standard CFG notation [18], let w denote the extrapolation weight between the conditional and dropped-condition branches. In our implementation, however, we use the common sampling-time parameterization $s = w + 1$, so that $s = 0$ corresponds to the null / unconditional branch, $s = 1$ recovers the plain conditional model, and $s > 1$ produces the usual CFG extrapolation. For comparison, an explicit classifier-guidance rule with the same scalar coefficient w appears in the VE sampler update as

$$\mathbf{d}_{\text{guide}} = \mathbf{d}_{\text{base}} - w\sigma\nabla_{\mathbf{x}_\sigma} \log p_\phi(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}). \tag{7}$$

Here $\mathbf{d}_{\text{guide}}$ and $\mathbf{d}_{\text{base}}$ denote the guided and unguided VE update fields, respectively. Thus, when w is used as an external guidance coefficient, the property gradient enters the sampler through a σ factor, making its effect noise-level dependent. Appendix A.1 gives the corresponding score-level derivation and the equivalent x_0-space form. In addition, explicit classifier guidance requires objective-specific calibration of both scale and direction. Different property targets can live on very different numeric ranges, such as QED in $[0, 1]$ versus Vina scores on a much wider scale, and some quantities are maximized while others, such as docking scores, are minimized. As a result, when w is used as an external guidance coefficient, its value and sign are generally not comparable across different property heads.

In contrast, CFG acts directly on the model predictions, which reduces the need for noise-level-dependent external-gradient calibration across different property metrics. Therefore, CFG induces the same linear extrapolation in the x_0 prediction, but without introducing a separate property predictor. For atom types, we apply the same CFG rule in the logit space of the categorical predictor, which is the standard practical counterpart of CFG for discrete outputs. Under this parameterization, this becomes

$$\begin{aligned} \hat{\mathbf{x}}_\theta^{\text{cfg}} &= D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing) + s(D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing)), \\ \hat{\mathbf{z}}_\theta^{\text{cfg}} &= \text{softmax}\Big(H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing) + s(H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing))\Big). \end{aligned} \tag{8}$$

Here the first line applies CFG to the coordinate prediction, while the second applies the same extrapolation to the type-logit branch before the softmax. We provide the corresponding score-level derivation and its relation to standard CFG formulas in Appendix A.2. Larger s makes the model follow the auxiliary property condition more strongly, while smaller s keeps the sample closer to the property-unconditional distribution under the same pocket.

3.5 Adaptive protein perturbation

The protein pocket is not treated as a perfectly clean ground truth during training. Experimental protein structures are subject to measurement error and capture only a static snapshot of a dynamic conformational ensemble. To account for this, we apply a small perturbation operator

$$\tilde{\mathcal{P}} = \mathrm{Perturb}(\mathcal{P}; \sigma_p), \tag{9}$$

where σ_p controls the scale of Gaussian coordinate perturbation on the pocket atoms. We train the denoiser on $\tilde{\mathcal{P}}$ instead of $\mathcal{P}$. This perturbation regularizes against over-reliance on a single crystallographic structure and encourages a neighborhood-aware conditional distribution. Following the classical connection between input noise and smoothness regularization [3, 5], a second-order expansion of a smooth denoising loss around the clean pocket shows that Gaussian pocket perturbation adds a leading-order penalty on the denoiser's sensitivity to pocket coordinates. For a squared denoising loss, this leading term contains $\frac{\sigma_p^2}{2}\|J_{\mathcal{P}} f_\theta\|_F^2$, where $J_{\mathcal{P}} f_\theta$ is the Jacobian of the denoising prediction with respect to the pocket input. Thus, the perturbation discourages sharp changes in the coordinate and type predictions under small pocket-coordinate variations, while still preserving the pocket as the conditioning signal. For the main results, we use a bounded noise-level-dependent schedule $\sigma_p = \min(0.1\sigma, 0.5)$. A formal statement and derivation are provided in Appendix E.

3.6 Full objective

The final training objective is

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathcal{L}_0, \mathcal{P}, \sigma, \tilde{\mathcal{P}}}\left[\lambda_x(\sigma)\|\hat{\mathbf{x}}_0^{\tilde{\mathcal{P}}} - \mathbf{x}_0\|_2^2 - \lambda_z(\sigma)\sum_{i=1}^{N_L}\sum_{k=1}^{K}(\mathbf{z}_0)_{ik}\log(\hat{\mathbf{z}}_0^{\tilde{\mathcal{P}}})_{ik}\right], \tag{10}$$

where $\hat{\mathbf{x}}_0^{\tilde{\mathcal{P}}}$ and $\hat{\mathbf{z}}_0^{\tilde{\mathcal{P}}}$ denote the predictions calculated using the perturbed pocket $\tilde{\mathcal{P}}$. At test time, the same model can be run either unguided or with classifier-free guidance, which gives a steerable trade-off between fidelity and target-specific preference.

4 Experiments

4.1 Experimental setup

Dataset We train our models on the CrossDocked2020 dataset [10]. Following prior work [31, 34, 14], we filter out binding poses with an RMSD > 1 Å and remove protein pairs with sequence identity $> 30\%$, resulting in a high-quality subset common in target-aware 3D molecule generation. This split is the standard benchmark for target-aware 3D generation, but docking-centered evaluation alone does not fully characterize whether generated ligands are geometrically plausible. To address this, we further adopt the GenBench3D protocol [1], which is designed to assess 3D conformation quality and check whether molecules remain physically plausible under stronger conditional guidance.

Baselines We compare **PocketVE** against representative target-aware 3D molecule generation baselines, including AR [31], Pocket2Mol [34], TargetDiff [14], DecompDiff [15], IPDiff [21], PAFlow [49], SeFMol [48], ALiDiff [13], PocketXMol [33], and BindDM [20]. **TAGMol** [7] is the architectural parent and ablation reference because it shares the base equivariant denoiser with PocketVE; this comparison reflects a bundled system transition rather than an isolated VE effect. We use TargetDiff as the primary established geometry comparison and PAFlow as the strong-affinity comparison that exposes the affinity–geometry trade-off. For baselines with released generated molecules, we re-evaluate those outputs under the same test split, docking protocol, and GenBench3D pipeline used for PocketVE.

Metrics We evaluate generated molecules on 100 test proteins, sampling 100 molecules per protein, and report both mean and median results. For binding affinity, we use AutoDock Vina [8] under the common setup of Luo et al. [31] and Ragoza et al. [35], reporting Vina Score, Vina Min, and Vina Dock. We also report High Affinity, the per-pocket fraction of generated molecules whose Vina Dock score is no worse than the reference ligand, and Joint-Ref, the fraction that simultaneously satisfies

no-worse Vina Dock, QED, and SA than the reference ligand. For molecular properties, we report QED [2], normalized SA [9], and diversity. For geometry, we report Valid_{3D}, clash-free rate, strain energy, and centroid distance under the GenBench3D protocol [1]. We also include TargetDiff-style distribution diagnostics based on Jensen-Shannon Divergence (JSD) over empirical ligand distance and atom-type statistics [14]. These JSD metrics are complementary distribution-fidelity diagnostics rather than explicit physical-validity metrics.

Implementation details PocketVE uses the target-aware equivariant network architecture of TAGMol [7] as the base denoiser, while modifying the diffusion parameterization, guidance mechanism, sampling procedure, and optimizer. For conditional control, we use training-set labels for Vina, QED, and normalized SA, discretize each property independently into five percentile bins, and apply joint condition dropout with probability 0.5 during training. In the main conditional experiments, the sampling condition uses the most favorable target bin for each property, namely the lowest Vina bin and the highest QED and SA bins. We train for $400{,}000$ optimization steps with batch size 4, validate every $2{,}000$ steps, and report the checkpoint with the lowest validation loss. For optimization, we use Muon [26] for hidden two-dimensional weight matrices and AdamW [29] for the remaining parameters, with learning rates 10^{-3} and 10^{-4}, respectively. Both learning rates use a plateau-based schedule: if the validation loss does not improve for $20{,}000$ steps, the learning rate is decayed by a factor of 0.6. For the main results, we use the noise-level-dependent pocket perturbation schedule $\sigma_p = \min(0.1\sigma, 0.5)$; alternative perturbation schedules are studied in Appendix Table 4. All main results use a 100-step sampler with the generalized arcsin schedule described in Appendix C. Across CFG scales, we use the same trained checkpoint and sampling hyperparameters, so changing the guidance scale only affects inference-time control. We provide the code for training, sampling, and evaluation in the supplementary material.

4.2 Main Results

Table 1: Summary of binding affinity, molecular properties, geometric quality, and diversity for reference molecules and molecules generated by **PocketVE** ($s = 5$) and other baselines. (↑)/(↓) denotes a larger / smaller number is better. Top 2 results are highlighted with **bold text** and underlined text, respectively. PocketVE is reported from one standardized run (100 pockets × 100 ligands), which is one of the three runs summarized in Appendix Table 7. TAGMol denotes the guided variant; the unguided variant appears in Table 2. A dash denotes an unavailable metric.

Method	Vina Score (↓)		Vina Min (↓)		Vina Dock (↓)		High	Joint	QED	SA	Valid_{3D}	Strain	Clash	Centroid	Diversity
	Avg.	Med.	Avg.	Med.	Avg.	Med.	Aff. (↑)	Ref. (↑)	Med. (↑)	Med. (↑)	(↑)	(↓)	Free (↑)	(↓)	Med. (↑)
Ref	-6.36	-6.46	-6.71	-6.49	-7.45	-7.26	-	-	0.47	0.74	75	65.3	97	-	-
AR	-5.75	-5.64	-6.18	-5.88	-6.75	-6.62	41.2	4.6	0.50	0.63	42.6	279.1	93.14	1.86	0.70
Pocket2Mol	-5.14	-4.70	-6.42	-5.82	-7.15	-6.79	48.0	16.5	0.57	0.75	57.3	130.6	81.03	1.82	0.71
TargetDiff	-5.47	-6.30	-6.64	-6.83	-7.80	-7.91	57.6	5.7	0.48	0.58	78.4	306.0	86.41	1.49	0.71
DecompDiff	-5.67	-6.04	-7.04	-7.09	-8.39	-8.43	63.8	6.8	0.43	0.60	71.5	318.4	78.00	2.90	0.68
IPDiff	-6.42	-7.01	-7.45	-7.48	-8.57	-8.51	68.2	10.0	0.53	0.59	52.0	1248.7	91.27	1.48	0.73
TAGMol	-7.02	-7.77	-7.95	-8.07	-8.59	-8.69	68.6	7.1	0.56	0.56	58.6	457.4	93.21	1.54	0.70
ALiDiff	-7.07	-7.95	-8.09	-8.17	-8.90	-8.81	68.5	7.0	0.50	0.56	54.7	1239.4	90.85	1.47	0.71
SeFMol	-7.23	-7.70	-8.03	-8.00	-8.72	-8.75	68.7	11.2	**0.64**	0.60	65.2	888.6	93.90	1.51	0.68
PocketXMol	-5.87	-6.14	-6.85	-6.79	-7.57	-7.59	-	-	0.51	**0.80**	67.0	**80.0**	93.46	1.79	**0.76**
BindDM	-	-	-7.29	-7.34	-8.41	-8.37	64.2	6.9	0.52	0.58	62.0	917.5	86.60	1.48	0.74
PAFlow	**-8.31**	**-8.92**	**-8.79**	**-8.96**	**-9.46**	**-9.49**	**80.8**	8.9	0.50	0.57	47.4	1834.6	**96.46**	1.55	0.70
Ours	-7.44	-7.89	-7.93	-8.19	-8.72	-8.89	73.9	**26.4**	**0.64**	0.71	**80.6**	127.9	94.02	**1.26**	0.71

We quantify the primary geometry comparison with a 10,000-replicate paired cluster bootstrap over complete pockets, using identical resampled pocket IDs for each method pair. Relative to TargetDiff, PocketVE maintains comparable Valid_{3D} ($+2.25$ percentage points; 95% CI $[-0.75, 5.13]$) while substantially improving geometric stability. The corresponding strain reduction, defined as TargetDiff minus PocketVE, is 178.10 (95% CI $[150.38, 203.44]$), making lower strain the clearest source-aligned gain. Against guided TAGMol, the paired difference in mean Vina Score is 0.43 in favor of PocketVE, but its 95% CI $[-0.02, 1.07]$ includes zero, so we do not claim a strict docking improvement over the architectural parent.

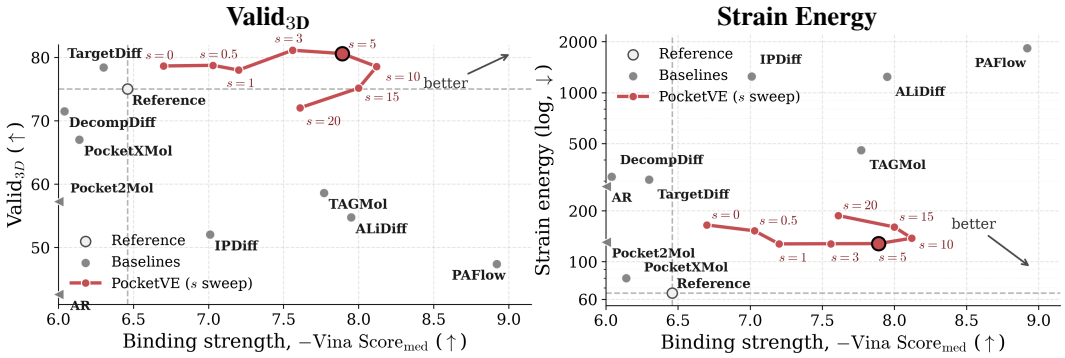


Figure 2: Binding–geometry trade-offs across baselines and PocketVE guidance scales. Both panels use $-\text{Vina Score}_{\text{med}}$ on the x-axis, so larger values indicate stronger predicted binding. Left: Valid_{3D}, where higher is better. Right: strain energy on a log scale, where lower is better. AR and Pocket2Mol lie outside the plotted x-axis range because their median Vina scores are substantially weaker (-5.64 and -4.70, respectively), and are therefore shown as off-scale markers at the left boundary. PocketVE traces a steerable operating curve that preserves strong geometric quality under moderate CFG scales.

JSD Diagnostics

Method	Local ↓	12Å↓	CC-2Å↓	Atom ↓
TAGMol	0.268	0.050	0.218	0.083
TargetDiff	0.234	0.056	0.213	0.059
Pocket2Mol	0.384	0.118	0.355	0.092
ALiDiff	0.341	0.097	0.350	0.202
IPDiff	0.391	0.098	0.349	0.194
PAFlow	0.454	0.141	0.461	0.228
Ours ($s = 0$)	**0.191**	**0.035**	**0.120**	**0.047**
Ours ($s = 5$)	0.255	0.065	0.269	0.120
Ours ($s = 10$)	0.284	0.067	0.292	0.136
Ours ($s = 20$)	0.326	0.068	0.306	0.124

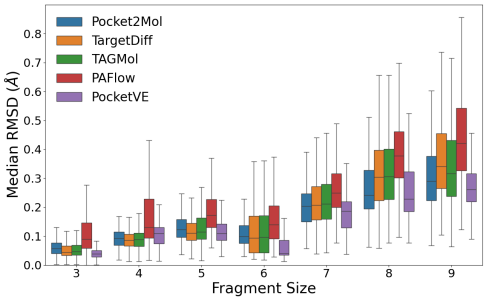


Figure 3: Additional geometric-fidelity analyses. Left: JSD-based distribution diagnostics adapted from TargetDiff [14], where generated molecules are compared against reference-ligand distributions from the test set. Lower values indicate better distribution fidelity. Right: Median RMSD between pre- and post-MMFF relaxed rigid fragments across fragment sizes for representative baselines.

Figure 2 makes the binding–geometry trade-off more explicit. Several strong-affinity baselines achieve low Vina scores but suffer from reduced Valid_{3D} or high strain. In contrast, moderate CFG in PocketVE improves predicted binding while staying close to the high-validity, low-strain regime. PAFlow illustrates this trade-off most clearly: it ranks first on the Vina-based metrics, but its much lower Valid_{3D} and substantially higher strain energy suggest that the predicted affinity gain comes with a large geometric cost. The corresponding QED–validity and SA–validity trade-off plots are provided in Appendix K.

Beyond the aggregate GenBench3D metrics, Figure 3 examines whether the geometric improvement is also visible at the distribution and fragment levels. The JSD diagnostics show that unguided PocketVE most closely matches the reference ligand distance and atom-type distributions, while increasing the CFG scale progressively moves samples away from the reference distribution. The rigid-fragment MMFF analysis gives a local check: compared with TargetDiff, Pocket2Mol, PAFlow, and TAGMol, PocketVE yields lower post-relaxation RMSD across most fragment sizes, especially for medium-to-large fragments, indicating more stable local geometry.

Figure 4 provides a representative visual comparison of the same trend. In this example, PocketVE places the ligand closer to the reference pose while maintaining a compact structure inside the pocket. Several baseline samples show larger pose deviations or less consistent pocket occupancy.

In summary, PocketVE improves geometric quality while retaining competitive docking and molecular-property scores rather than uniformly dominating every metric (Table 1). Figures 2, 3, and 4 provide aggregate, distributional, fragment-level, and qualitative views of this trade-off. Moderate CFG preserves geometric stability while shifting target-related properties, whereas stronger

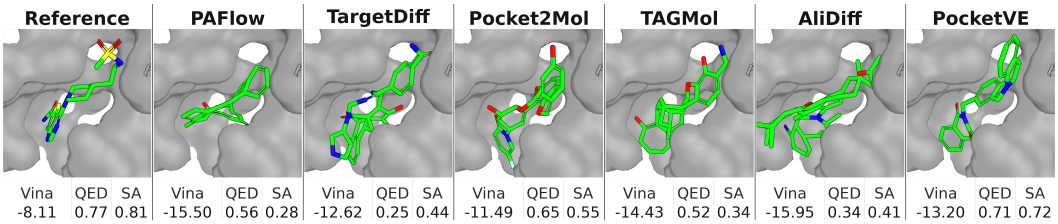


Figure 4: Visualizations of the reference ligand and generated ligands from baselines and PocketVE for a representative protein pocket (PDB ID: 5D7N). Vina Dock, QED, and SA are reported below.

Table 2: Compact ablation and efficiency summary for PocketVE. The left table reports a cumulative ablation from TAGMol to the final PocketVE configuration: the PocketVE backbone row uses $s = 0$, no protein perturbation, the log-uniform schedule, and the default sampling-time noise injection, and each subsequent row adds one component on top of the previous row. Runtimes are measured in seconds to sample 100 ligands per protein over 100 test proteins on a single NVIDIA A100 GPU with batch size 10 and no mixed precision, excluding docking, MMFF relaxation, and other post-processing.

Cumulative Ablation

Setting	Vina Med. ↓	Valid$_{3D}$ ↑	Strain ↓	Clash-Free ↑
TAGMol (unguided)	-6.08	65.56	401.3	78.13
PocketVE backbone	-6.35	69.62	274.4	88.59
+ gen-arcsin scheduler	-6.76	77.75	179.1	91.87
+ CFG scale $s = 1$	-7.41	78.55	137.8	92.70
+ CFG scale $s = 5$	-7.78	77.33	**125.9**	**94.07**
+ protein perturb.	**-7.89**	**80.60**	127.9	94.02

Sampling Time

Method	Time (↓)
PocketVE ($s = 1$)	**127.8**
PocketVE ($s = 5$)	234.1
PAFlow	249.8
TargetDiff	1382.5
Pocket2Mol	1396.8
TAGMol	2433.3

guidance can increase distribution drift and reduce parts of the diversity profile. Appendix N reports pocket-permutation and PoseCheck diagnostics, and Appendices I and J provide repeated-run and reference-normalized analyses.

4.3 Ablations and Analysis

We ablate the cumulative effect of the main components (Table 2), followed by two control axes that preserve the base architecture: the inference-time CFG scale and the training-time protein perturbation strategy.

Cumulative ablation The left panel of Table 2 shows that the PocketVE backbone and sampler yield a large geometric improvement over TAGMol, while moderate CFG mainly shifts affinity-related objectives. This comparison reflects an integrated transition in parameterization, scaling, noise, and sampling, rather than an isolated test of VE alone. Protein perturbation then improves the final geometry-oriented operating point, raising Valid$_{3D}$ while keeping other metrics nearly unchanged. A component-wise summary that groups the TAGMol baseline, unguided PocketVE, CFG variants, protein-perturbation variants, and schedule ablation is provided in Appendix H.

As a targeted coordinate-scale check, we also removed the σ_{data} factor in Eq. (32). In the corresponding diagnostic, mean Vina changes from -7.589 to $+92.237$, clash-free rate collapses from 94.23% to 3.39%, and centroid distance increases from 1.25 to 3.33 Å, while Valid$_{3D}$ remains 78.72% versus 82.39%. We therefore retain Eq. (32) as part of the integrated pocket-aware formulation, without attributing the full gain to this factor alone.

Sampling efficiency The PocketVE ($s = 5$) runtime on the right already includes the extra CFG passes and uses the same 100-step regime as PAFlow, VEDA [46], and SemlaFlow [22]. Under the same evaluation protocol, PocketVE remains substantially faster than most diffusion and autoregressive baselines and is slightly faster than PAFlow while remaining in the same runtime regime. The runtime comparison on the right should be read together with the sampling-step ablation in Appendix D. In our setting, the 100-step sampler is a deliberate operating point supported by that ablation, rather than a naive truncation of a longer trajectory.

Guidance scale sensitivity We sweep the sampling-time CFG scale s, where $s = 0$ uses the null/unconditional branch and $s = 1$ recovers the plain conditional model. As shown in Figure 2, increasing s first improves target-related objectives while largely preserving geometric quality. However, overly large guidance scales eventually reduce Valid_{3D} and increase distribution drift. In our experiments, $s = 5$ is the representative operating point, whereas $s = 10$ further favors affinity-related objectives at a geometric cost. Detailed step-wise metrics and per-metric trend plots are provided in Appendix F.

We also evaluate property distributions, non-default requests, and conflicting conditions in Appendix O. These analyses show directional property shifts under CFG together with trade-offs in geometry, distributional fidelity, and diversity at stronger guidance scales.

Protein perturbation strategy To evaluate the impact of training-time protein-side perturbation, we compared fixed and noise-level-dependent schedules against a no-perturbation baseline. Protein perturbation acts as a regularizer that reshapes the property–geometry trade-off rather than uniformly dominating the no-perturbation baseline. Among the evaluated settings, the bounded noise-level-dependent schedule $\sigma_p = \min(0.1\sigma, 0.5)$ gives the most favorable trade-off across local JSD, atom-type JSD, Vina metrics, and centroid distance. Lighter perturbation further reduces strain, while stronger schedules improve broader distributional statistics such as JSD-All-12Å, JSD-CC-2Å, and SA at some cost in target-specific precision. Detailed JSD diagnostics, full GenBench3D metrics, and an additional test-time pocket perturbation study are provided in Appendix Tables 4 and 9.

5 Limitations

Our method has several limitations that affect its practical deployment. First, classifier-free guidance increases inference cost because each guided sampling step requires both a conditional and an unconditional forward pass. This doubles the number of function evaluations compared with plain conditional sampling. Second, the preferred guidance scale may be target-dependent, and stronger guidance can trade geometric and distributional fidelity for target-related objectives. Third, the Gaussian pocket perturbation used here is a simple training regularizer rather than a realistic model of protein conformational flexibility. Finally, although PocketVE improves geometry on the evaluated benchmark, it relies on docking-based affinity proxies and benchmark-specific protocols. All principal results use the filtered 100-pocket CrossDocked2020 split, so broader generalization and experimental validation remain future work. The TAGMol-to-PocketVE transition also combines several backbone and sampler choices, and the current experiments do not establish VE-only causal attribution. Additionally, cross-method comparison is confounded by differences in generated atom count (molecular size), and all results use first-order Euler sampling, which is not claimed to be the optimal solver. We use $\sigma_{\text{data}} = 10$ as a fixed operating choice; the sensitivity sweep in Appendix Q shows that performance remains robust across the evaluated range. Vina-based affinity proxies may reward over-optimization of binding geometry rather than true binding affinity, and the guidance scale $s = 5$ was selected descriptively from the test-benchmark sweep rather than from an independent validation set.

6 Conclusion

In this work, we introduced PocketVE, a target-aware 3D molecular generation framework that improves geometric stability while supporting inference-time conditional control. By revisiting coordinate parameterization, guidance design, and protein-side perturbation, PocketVE achieves higher structural quality in unguided generation and a more favorable trade-off between geometry and target objectives under guidance. Experiments on CrossDocked2020 show improved geometric stability relative to most baselines, together with directional, aggregate multi-property steering. Future work includes extending evaluation to broader target families, introducing matched component controls, and strengthening the connection between computational diagnostics and experimental validation.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Institutes of Health (NIH) under Grants R01CA297855, U19AI171421, and R01DA051922, and by the U.S. National Science Foundation (NSF) under Grant AGS-2426655. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH or NSF. The authors declare no competing interests.

References

[1] Benoit Baillif, Jason Cole, Patrick McCabe, and Andreas Bender. Benchmarking structure-based three-dimensional molecular generative models using genbench3d: ligand conformation quality matters. *arXiv preprint arXiv:2407.04424*, 2024.

[2] G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

[3] Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

[4] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: enabling multimodal flows with applications to protein co-design. In *Proceedings of the 41st International Conference on Machine Learning*, pages 5453–5512, 2024.

[5] Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. *Advances in neural information processing systems*, 13, 2000.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[7] Vineeth Dorna, D Subhalingam, Keshav Kolluru, Shreshth Tuli, Mrityunjay Singh, Saurabh Singal, NM Anoop Krishnan, and Sayan Ranu. Tagmol: Target-aware gradient-guided molecule generation. In *ICML'24 Workshop ML for Life and Material Science: From Theory to Industry Applications*.

[8] Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. Autodock vina 1.2. 0: new docking methods, expanded force field, and python bindings. *Journal of chemical information and modeling*, 61(8):3891–3898, 2021.

[9] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):8, 2009.

[10] Paul G Francoeur, Tomohide Masuda, Jocelyn Sunseri, Andrew Jia, Richard B Iovanisci, Ian Snyder, and David R Koes. Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design. *Journal of chemical information and modeling*, 60(9):4200–4215, 2020.

[11] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[12] Colin R Groom, Ian J Bruno, Matthew P Lightfoot, and Suzanna C Ward. The cambridge structural database. *Structural Science*, 72(2):171–179, 2016.

[13] Siyi Gu, Minkai Xu, Alexander Powers, Weili Nie, Tomas Geffner, Karsten Kreis, Jure Leskovec, Arash Vahdat, and Stefano Ermon. Aligning target-aware molecule diffusion models with exact energy optimization. *Advances in Neural Information Processing Systems*, 37:44040–44063, 2024.

[14] Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. In *The Eleventh International Conference on Learning Representations*.

[15] Jiaqi Guan, Xiangxin Zhou, Yuwei Yang, Yu Bao, Jian Peng, Jianzhu Ma, Qiang Liu, Liang Wang, and Quanquan Gu. Decompdiff: diffusion models with decomposed priors for structure-based drug design. In *Proceedings of the 40th International Conference on Machine Learning*, pages 11827–11846, 2023.

[16] Charles Harris, Kieran Didi, Arian R Jamasb, Chaitanya K Joshi, Simon V Mathis, Pietro Lio, and Tom Blundell. Benchmarking generated poses: How rational is structure-based drug design with generative models? *arXiv preprint arXiv:2308.07413*, 2023.

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

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

[19] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pages 8867–8887. PMLR, 2022.

[20] Zhilin Huang, Ling Yang, Zaixi Zhang, Xiangxin Zhou, Yu Bao, Xiawu Zheng, Yuwei Yang, Yu Wang, and Wenming Yang. Binding-adaptive diffusion models for structure-based drug design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12671–12679, 2024.

[21] Zhilin Huang, Ling Yang, Xiangxin Zhou, Zhilong Zhang, Wentao Zhang, Xiawu Zheng, Jie Chen, Yu Wang, Bin Cui, and Wenming Yang. Protein-ligand interaction prior for binding-aware 3d molecule diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

[22] Ross Irwin, Alessandro Tibo, Jon Paul Janet, and Simon Olsson. Semlaflow–efficient 3d molecular generation with latent attention and equivariant flow matching. In *International Conference on Artificial Intelligence and Statistics*, pages 3772–3780. PMLR, 2025.

[23] Clemens Isert, Kenneth Atz, and Gisbert Schneider. Structure-based drug design with geometric deep learning. *Current Opinion in Structural Biology*, 79:102548, 2023.

[24] Yue Jian, Curtis Wu, Danny Reidenbach, and Aditi S Krishnapriyan. General binding affinity guidance for diffusion models in structure-based drug design. *Journal of Chemical Information and Modeling*, 2026.

[25] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.

[26] Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024.

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

[28] Hannes H Loeffler, Jiazhen He, Alessandro Tibo, Jon Paul Janet, Alexey Voronov, Lewis H Mervin, and Ola Engkvist. Reinvent 4: Modern ai–driven generative molecule design. *Journal of Cheminformatics*, 16(1):20, 2024.

[29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

[30] Aaron Lou and Stefano Ermon. Reflected diffusion models. In *International Conference on Machine Learning*, pages 22675–22701. PMLR, 2023.

[31] Shitong Luo, Jiaqi Guan, Jianzhu Ma, and Jian Peng. A 3d generative model for structure-based drug design. *Advances in Neural Information Processing Systems*, 34:6229–6239, 2021.

[32] Filipp Nikitin, Ian Dunn, David Ryan Koes, and Olexandr Isayev. Geom-drugs revisited: toward more chemically accurate benchmarks for 3d molecule generation. *Digital Discovery*, 4(11):3282–3291, 2025.

[33] Xingang Peng, Ruihan Guo, Fenglin Guo, Ziyi Wang, Jiayu Sun, Jiaqi Guan, Yinjun Jia, Yan Xu, Yanwen Huang, Muhan Zhang, et al. Unified modeling of 3d molecular generation via atomic interactions with pocketxmol. *Cell*, 2026.

[34] Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2mol: Efficient molecular sampling based on 3d protein pockets. In *International conference on machine learning*, pages 17644–17655. PMLR, 2022.

[35] Matthew Ragoza, Tomohide Masuda, and David Ryan Koes. Generating 3d molecules conditional on receptor binding sites with deep generative models. *Chemical science*, 13(9):2701–2713, 2022.

[36] Kevin Rojas, Ye He, Chieh-Hsin Lai, Yuhta Takida, Yuki Mitsufuji, and Molei Tao. Improving classifier-free guidance in masked diffusion: Low-dim theoretical insights with high-dim impact. In *The Fourteenth International Conference on Learning Representations*, 2026.

[37] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[38] Arne Schneuing, Charles Harris, Yuanqi Du, Kieran Didi, Arian Jamasb, Ilia Igashov, Weitao Du, Carla Gomes, Tom L Blundell, Pietro Lio, et al. Structure-based drug design with equivariant diffusion models. *Nature Computational Science*, 4(12):899–909, 2024.

[39] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*.

[40] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.

[41] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. Ieee, 2015.

[42] Jiahui Tong and Suwen Zhao. Large-scale analysis of bioactive ligand conformational strain energy by ab initio calculation. *Journal of Chemical Information and Modeling*, 61(3):1180–1192, 2021.

[43] Clement Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. Midi: Mixed graph and 3d denoising diffusion for molecule generation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 560–576. Springer, 2023.

[44] Minkai Xu, Alexander S Powers, Ron O Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In *International Conference on Machine Learning*, pages 38592–38610. PMLR, 2023.

[45] Peining Zhang, Daniel Baker, Minghu Song, and Jinbo Bi. Unraveling the potential of diffusion models in small-molecule generation. *Drug Discovery Today*, 30(7):104413, 2025.

[46] Peining Zhang, Jinbo Bi, and Minghu Song. Veda: Generation of 3d molecules via variance-exploding diffusion with annealing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 28346–28354, 2026.

[47] Pengze Zhang, Hubery Yin, Chen Li, and Xiaohua Xie. Tackling the singularities at the endpoints of time intervals in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6945–6954, 2024.

[48] Xudong Zhang, Sanqing Qu, Fan Lu, Jianmin Wang, Zhixin Tian, Shangding Gu, Yanping Zhang, Alois Knoll, Shaorong Gao, Guang Chen, et al. Steering semi-flexible molecular diffusion model for structure-based drug design with reinforcement learning. *Science Advances*, 12(16):eady9955, 2026.

[49] Jingyuan Zhou, Hao Qian, Shikui Tu, and Lei Xu. Prior-guided flow matching for target-aware molecule design with learnable atom number. *arXiv preprint arXiv:2509.01486*, 2025.

[50] Zhijian Zhou, Junyi An, Zongkai Liu, Yunfei Shi, Xuan Zhang, Fenglei Cao, Chao Qu, and Yuan Qi. Guiding diffusion models with reinforcement learning for stable molecule generation. *arXiv preprint arXiv:2508.16521*, 2025.

A Classifier-Free Guidance Details

For completeness, we record the standard score-level form of classifier-free guidance used to motivate the implementation in the main text.

A.1 Noise-Level Dependence of Explicit Classifier Guidance

Let $\mathcal{L}_\sigma = (\mathbf{x}_\sigma, \mathbf{z}_\sigma)$ denote the noisy ligand state at noise level σ. The guided target distribution can be written as

$$\log p_w(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) = (1+w)\log p(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) - w \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing) + \text{const}. \tag{11}$$

For the continuous coordinate branch, taking the score with respect to $\mathbf{x}_\sigma$ yields

$$\nabla_{\mathbf{x}_\sigma} \log p_w(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) = (1+w)\nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) - w\nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing). \tag{12}$$

At the score level, both approaches can be read through the conditional decomposition

$$\nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) = \nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing) + \nabla_{\mathbf{x}_\sigma} \log p(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}), \tag{13}$$

which, when substituted into Eq. (12), gives

$$\begin{aligned} \nabla_{\mathbf{x}_\sigma} \log p_w(\mathcal{L}_\sigma \mid \mathcal{P}, \mathbf{a}) &= (1+w)\Big(\nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing) + \nabla_{\mathbf{x}_\sigma} \log p(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P})\Big) \\ &\quad - w\nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing) \\ &= \nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing) + (1+w)\nabla_{\mathbf{x}_\sigma} \log p(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}). \end{aligned} \tag{14}$$

This makes explicit that extrapolating the conditional score away from the dropped-condition score amplifies the same property-dependent term used by external guidance methods. Explicit classifier guidance follows the same decomposition, but replaces that attribute-dependent term with the gradient of an external classifier or property predictor. More concretely, it keeps the base generative score $s_{\text{base}}(\mathbf{x}_\sigma) \approx \nabla_{\mathbf{x}_\sigma} \log p(\mathcal{L}_\sigma \mid \mathcal{P}, \varnothing)$ and approximates the attribute-dependent term by $\nabla_{\mathbf{x}_\sigma} \log p_\phi(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}) \approx \nabla_{\mathbf{x}_\sigma} \log p(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P})$. Introducing the same scalar coefficient w as an external guidance strength then gives

$$s_{\text{guide}}(\mathbf{x}_\sigma) = s_{\text{base}}(\mathbf{x}_\sigma) + (1+w)\nabla_{\mathbf{x}_\sigma} \log p_\phi(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}), \tag{15}$$

where s_{base} denotes the generative score and p_ϕ denotes an external classifier or property predictor. When taking the score with respect to $\mathbf{x}_\sigma$, we treat $\mathbf{z}_\sigma$ as part of the conditioning context of the joint state. Under the VE parameterization, this relation follows from the standard Tweedie identity. The forward process is $\mathbf{x}_\sigma = \mathbf{x}_0 + \sigma\boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I)$, so $p(\mathbf{x}_\sigma \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma^2 I)$. For fixed conditioning context $(\mathbf{z}_\sigma, \mathcal{P}, \mathbf{a})$, Tweedie's formula gives

$$\mathbb{E}[\mathbf{x}_0 \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}, \mathbf{a}] = \mathbf{x}_\sigma + \sigma^2 \nabla_{\mathbf{x}_\sigma} \log p(\mathbf{x}_\sigma \mid \mathbf{z}_\sigma, \mathcal{P}, \mathbf{a}). \tag{16}$$

Approximating this posterior mean with the denoiser prediction $D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) \approx \mathbb{E}[\mathbf{x}_0 \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}, \mathbf{a}]$ and rearranging yields

$$\nabla_{\mathbf{x}_\sigma} \log p(\mathbf{x}_\sigma \mid \mathbf{z}_\sigma, \mathcal{P}, \mathbf{a}) \approx \frac{D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - \mathbf{x}_\sigma}{\sigma^2}. \tag{17}$$

Applying the same VE identity to the explicit classifier-guidance score in Eq. (15) yields the equivalent x_0-space correction

$$D_{\text{guide}}(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}) \approx D_{\text{base}}(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}) + w\sigma^2 \nabla_{\mathbf{x}_\sigma} \log p_\phi(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}). \tag{18}$$

Applying Eq. (17) to both the conditional and dropped-condition branches in Eq. (12) gives

$$\nabla_{\mathbf{x}_\sigma} \log p_w \approx \frac{(1+w)D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - wD_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing) - \mathbf{x}_\sigma}{\sigma^2}. \tag{19}$$

Under the probability-flow ODE parameterization of VE models [40, 27], the reverse update field induced by a clean prediction $\hat{\mathbf{x}}_0$ can be written as

$$\mathbf{d} = \frac{\mathbf{x}_\sigma - \hat{\mathbf{x}}_0}{\sigma}. \tag{20}$$

Applying this identity to the base predictor $D_{\text{base}}(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P})$ gives the unguided field

$$\mathbf{d}_{\text{base}} = \frac{\mathbf{x}_\sigma - D_{\text{base}}(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P})}{\sigma}. \tag{21}$$

Substituting the classifier-guided clean prediction from Eq. (18) then recovers the main-text update rule

$$\mathbf{d}_{\text{guide}} = \mathbf{d}_{\text{base}} - w\sigma \nabla_{\mathbf{x}_\sigma} \log p_\phi(\mathbf{a} \mid \mathbf{x}_\sigma, \mathbf{z}_\sigma, \mathcal{P}), \tag{22}$$

which is Eq. (7) in the main text. This makes the noise-level dependence explicit: an external classifier gradient enters the denoiser through a σ^2 factor, and the sampler update through a σ factor.

A.2 Recovering the Classifier-Free Guidance Rule

Returning to Eq. (12), we now recover the main-text implementation of classifier-free guidance. Comparing the expression above with the VE identity

$$\nabla_{\mathbf{x}_\sigma} \log p_w = \frac{\hat{\mathbf{x}}_\theta^{\mathrm{cfg}} - \mathbf{x}_\sigma}{\sigma^2}, \tag{23}$$

we identify the guided clean prediction as

$$\hat{\mathbf{x}}_\theta^{\mathrm{cfg}} = (1+w) D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - w D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing). \tag{24}$$

Using the implementation parameterization $s = w + 1$ (equivalently, $w = s - 1$) then yields

$$\hat{\mathbf{x}}_\theta^{\mathrm{cfg}} = D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing) + s(D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing)), \tag{25}$$

which is exactly the coordinate branch in Eq. (8) of the main text. For the discrete branch, we do not use a continuous score over $\mathbf{z}_\sigma$. Instead, we apply the same conditional / dropped-condition extrapolation directly to the categorical logits:

$$\hat{H}_\theta^{\mathrm{cfg}} = H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing) + s(H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \mathbf{a}) - H_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}, \varnothing)). \tag{26}$$

Applying $\mathrm{softmax}(\hat{H}_\theta^{\mathrm{cfg}})$ then recovers the discrete branch in Eq. (8).

B EDM Preconditioning and Target-Aware Coordinate Scaling

This section summarizes the EDM preconditioning recipe used by PocketVE and distinguishes the standard VE/EDM machinery from the target-aware adaptation introduced by our pocket-conditioned setting. The coefficient definitions themselves are standard EDM/Karras-style components reused from VEDA [27, 46]; the main PocketVE-specific point is how the ligand coordinates are scaled before entering the shared protein–ligand geometric graph.

Under the VE corruption process, clean ligand coordinates are perturbed as

$$\mathbf{x}_\sigma = \mathbf{x}_0 + \sigma \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I). \tag{27}$$

Standard EDM writes the coordinate denoiser in the preconditioned form

$$D_\theta(\mathbf{x}_\sigma, \sigma) = c_{\mathrm{skip}}(\sigma)\mathbf{x}_\sigma + c_{\mathrm{out}}(\sigma)\, F_\theta(c_{\mathrm{in}}(\sigma)\mathbf{x}_\sigma,\, c_{\mathrm{noise}}(\sigma)), \tag{28}$$

with

$$c_{\mathrm{skip}}(\sigma) = \frac{\sigma_{\mathrm{data}}^2}{\sigma^2 + \sigma_{\mathrm{data}}^2}, \qquad c_{\mathrm{out}}(\sigma) = \frac{\sigma\, \sigma_{\mathrm{data}}}{\sqrt{\sigma^2 + \sigma_{\mathrm{data}}^2}}, \tag{29}$$

$$c_{\mathrm{in}}(\sigma) = \frac{1}{\sqrt{\sigma^2 + \sigma_{\mathrm{data}}^2}}, \qquad c_{\mathrm{noise}}(\sigma) = \frac{\log \sigma}{4}. \tag{30}$$

Here c_{in} normalizes the noisy input magnitude, c_{skip} keeps the prediction anchored to the noisy sample at low noise, c_{out} rescales the network residual back to the clean-coordinate domain, and c_{noise} provides the scalar noise embedding.

PocketVE keeps the standard output preconditioning and noise embedding, while modifying how the EDM input scaling is applied before the pocket–ligand graph. At the notation level used in the main text, this is written as

$$D_\theta(\mathbf{x}_\sigma, \mathbf{z}_\sigma, \sigma, \mathcal{P}) = c_{\mathrm{skip}}(\sigma)\mathbf{x}_\sigma + c_{\mathrm{out}}(\sigma)\, F_\theta(\tilde{\mathbf{x}}_\sigma;\, c_{\mathrm{noise}}(\sigma),\, \mathbf{z}_\sigma,\, \mathcal{P}). \tag{31}$$

In the code, this is realized through an equivalent two-stage implementation: the network first operates on the scaled ligand coordinates

$$\mathbf{x}_{\mathrm{net}} = \sigma_{\mathrm{data}}\, c_{\mathrm{in}}(\sigma)\mathbf{x}_\sigma = \tilde{\mathbf{x}}_\sigma, \tag{32}$$

and predicts the coordinates in that scaled frame as

$$\hat{\mathbf{x}}_{\mathrm{net}} = F_\theta(\mathbf{x}_{\mathrm{net}};\, c_{\mathrm{noise}}(\sigma),\, \mathbf{z}_\sigma,\, \mathcal{P}), \tag{33}$$

and the final coordinate prediction in the default constant mode is reconstructed as

$$\hat{\mathbf{x}}_0 = c_{\text{skip}}(\sigma)\mathbf{x}_\sigma + c_{\text{out}}(\sigma)\big(\hat{\mathbf{x}}_{\text{net}} - \mathbf{x}_{\text{net}}\big). \tag{34}$$

This makes explicit that the network prediction is interpreted relative to the scaled ligand input before being mapped back to the original coordinate frame. The discrete atom-type branch is handled separately through the DFM-based categorical predictor, so the present discussion is only about the continuous coordinate preconditioning.

The target-aware adaptation appears in the network input. In textbook EDM, one would feed $c_{\text{in}}(\sigma)\mathbf{x}_\sigma$ into the coordinate backbone. In the actual PocketVE implementation, however, the ligand coordinates are effectively fed as

$$\sigma_{\text{data}}\, c_{\text{in}}(\sigma)\mathbf{x}_\sigma = \frac{\sigma_{\text{data}}}{\sqrt{\sigma^2 + \sigma_{\text{data}}^2}}\mathbf{x}_\sigma = \tilde{\mathbf{x}}_\sigma. \tag{35}$$

The same scaled input is used as the reference point in the output reconstruction, so the model predicts a correction relative to the scaled ligand coordinates rather than absolute coordinates in the network frame. The design is necessary because ligand and protein coordinates interact inside the same distance-based equivariant graph, while the pocket coordinates are not scaled by EDM coefficients. If the ligand alone were shrunk by the textbook $c_{\text{in}}(\sigma)$ factor, then at low noise the ligand would be rescaled by approximately $1/\sigma_{\text{data}}$ while the pocket remained in the original coordinate frame, distorting protein–ligand distances. The scale-preserving input together with the matching output recentering avoids that mismatch. As $\sigma \to 0$, it satisfies $\tilde{\mathbf{x}}_\sigma \to \mathbf{x}_\sigma$, so the ligand remains in the same physical frame as the pocket near the clean limit, while at larger noise levels it still attenuates the coordinate magnitude in the same spirit as EDM.

This target-aware scaling also explains why PocketVE uses a comparatively large σ_{data}. Our coordinates are normalized around the pocket center rather than around each ligand itself, so σ_{data} must cover both the internal ligand extent and the ligand displacement from the pocket center. For the same reason, we keep $\sigma_{\min} = 10^{-3}$ small enough to preserve near-clean coordinate precision in the final reverse steps. For training, the continuous coordinate branch uses the EDM-style weighting induced by the implementation loss $\|\hat{\mathbf{x}}_0 - \mathbf{x}_0\|_2^2 / c_{\text{out}}(\sigma)^2$, while the categorical branch uses a separately weighted cross-entropy term. The concrete dataset statistics behind these choices, together with the full schedule specification, are given in Appendix C.

C Sampling Details

Training was run on a single NVIDIA A100 GPU for approximately 24 hours. The host machine used four AMD EPYC 7513 32-Core Processor CPUs and 100 GB of memory.

For the VE coordinate branch, we set $\sigma_{\max} = 800$, $\sigma_{\min} = 10^{-3}$, and $\sigma_{\text{data}} = 10.0$. The choice of $\sigma_{\text{data}} = 10.0$ is target-aware rather than ligand-only. On the training set, the per-ligand coordinate spread has mean and median standard deviation 3.82 Å, with a maximum of 9.91 Å. Because our coordinates are centered by the protein pocket rather than by each ligand itself, the relevant clean-data scale must also cover ligand displacement from the pocket center. Under this normalization, the ligand-center distance has mean 2.31 Å, median 1.98 Å, and maximum 9.64 Å. We therefore choose $\sigma_{\text{data}} = 10.0$ so that the preconditioning scale covers both the internal ligand extent and the pocket-relative ligand offset. We also keep $\sigma_{\min} = 10^{-3}$ small enough to preserve near-clean coordinate precision at the end of the reverse trajectory.

We traverse the noise levels with a generalized arcsin schedule, which extends the arcsin scheduler used in VEDA [46]. Let $\tau_i = 1 - i/N$ and

$$\sigma_i = \exp\left(\log \sigma_{\min} + \left[(1-\beta)\tau_i + \beta\frac{2}{\pi}\arcsin(\tau_i^p)\right](\log \sigma_{\max} - \log \sigma_{\min})\right), \tag{36}$$

where we use $\beta = 2.2$ and $p = 0.57$ in all experiments. Let $\sigma_0 > \sigma_1 > \cdots > \sigma_N$ denote the resulting noise schedule.

For the discrete atom types, we first convert the type head into a clean categorical prediction

$$\mathbf{p}_{\theta,i} = \text{softmax}\big(H_\theta(\mathbf{x}_{\sigma_i}, \mathbf{z}_{\sigma_i}, \sigma_i, \mathcal{P})\big). \tag{37}$$

We then update $\mathbf{z}_{\sigma_i}$ with a discrete sampler based on the DFM transition-rate formulation [4]. In this view, the transition rate toward category j is

$$R_\theta(z_{\sigma_i}, j) = \omega(\sigma_i)\, p_\theta(z_0 = j \mid z_{\sigma_i}) + \eta_i\, p_\theta(z_0 = z_{\sigma_i} \mid z_{\sigma_i}), \tag{38}$$

where

$$\omega(\sigma_i) = \frac{\eta_i S(1 - m(\sigma_i)) + \eta_i m(\sigma_i) + m'(\sigma_i)}{m(\sigma_i)}, \tag{39}$$

S is the number of atom categories, $p_\theta(z_0 = \cdot \mid z_{\sigma_i})$ is given by the predicted categorical distribution from Eq. (37), and we follow the variable-noise setting in [4] with $\eta_i = \eta / m'(\sigma_i)$. Equivalently, this step can be interpreted as masked-token refinement under the same schedule $m(\sigma)$.

D Sampling-Step Ablation

We ablate the number of reverse sampling steps using representative CFG scales $s \in \{0, 1, 5, 10\}$. Figure 5 shows that moving from 25 to 100 steps improves both predicted binding strength and 3D validity across guidance scales, while increasing from 100 to 200 steps gives smaller additional gains relative to the extra sampling cost. We therefore use $N = 100$ as the default operating point in the main experiments.

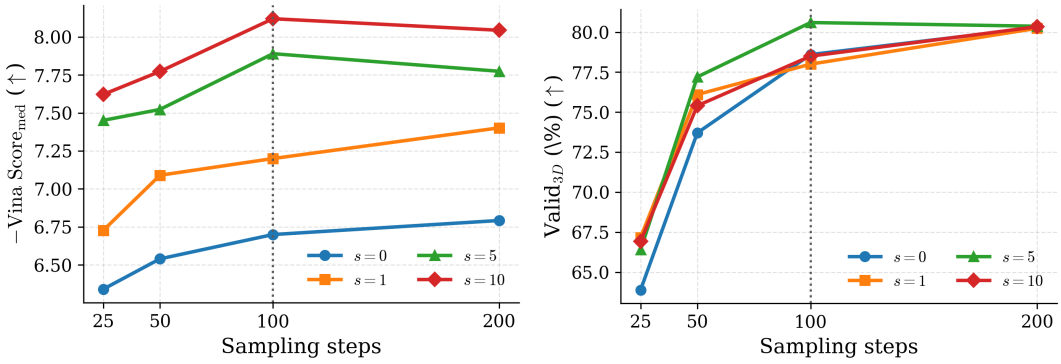


Figure 5: Sampling-step ablation under representative CFG scales. Left: median Vina Score is shown as $-\text{Vina Score}_{\text{med}}$, so higher is better. Right: Valid_{3D} under the GenBench3D protocol. The vertical dashed line marks the default setting $N = 100$.

Table 3: Sampling-step ablation for representative CFG scales. We report median Vina Score, Valid_{3D}, and median strain energy. Lower Vina Score and strain energy are better; higher Valid_{3D} is better.

s	Steps	Vina Score (↓)	Valid_{3D} (↑)	Strain (↓)
0	25	-6.34	63.89	398.3
0	50	-6.54	73.70	231.3
0	100	-6.70	78.60	164.9
0	200	-6.79	80.29	144.4
1	25	-6.73	67.17	324.1
1	50	-7.09	76.10	178.9
1	100	-7.20	78.00	127.4
1	200	-7.40	80.24	116.5
5	25	-7.45	66.40	293.0
5	50	-7.52	77.20	158.8
5	100	-7.89	**80.60**	127.9
5	200	-7.77	80.37	**111.2**
10	25	-7.62	66.93	326.5
10	50	-7.77	75.40	172.7
10	100	**-8.12**	78.50	137.7
10	200	-8.04	80.34	118.9

E Pocket Perturbation Details

In our implementation, pocket perturbation is applied directly to pocket coordinates:

$$\tilde{\mathbf{y}} = \mathbf{y} + \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma_p^2 \mathbf{I}), \tag{40}$$

while pocket atom features are kept unchanged.

The following local statement formalizes the regularization view of pocket perturbation using the standard connection between input noise, vicinal risk minimization, and Jacobian regularization [3, 5].

Proposition 1 (Pocket perturbation regularizes conditional sensitivity)**.** *Let $f_\theta(\cdot, \mathcal{P})$ denote either the coordinate denoising head or the pre-softmax type head at a fixed ligand noise level σ, and let $\ell(f_\theta(\cdot, \mathcal{P}), t)$ be a smooth per-sample denoising loss with target t. For Gaussian pocket perturbation $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma_p^2 I)$, the perturbed-pocket objective satisfies*

$$\mathbb{E}_{\boldsymbol{\eta}}\, \ell(f_\theta(\cdot, \mathcal{P} + \boldsymbol{\eta}), t) = \ell(f_\theta(\cdot, \mathcal{P}), t) + \frac{\sigma_p^2}{2} \mathrm{Tr}\big(\nabla^2_{\mathcal{P}} \ell(f_\theta(\cdot, \mathcal{P}), t)\big) + O(\sigma_p^4). \tag{41}$$

under standard smoothness assumptions. For a squared coordinate loss, the leading nonnegative term contains $\frac{\sigma_p^2}{2} \|J_{\mathcal{P}} f_\theta\|_F^2$ near the optimum. Thus pocket perturbation penalizes sharp dependence of the denoiser on small pocket-coordinate changes. Equivalently, the model is trained against a local neighborhood of mildly perturbed pocket views rather than a single rigid conditioning instance. When the perturbation scale is small, this neighborhood can be interpreted as approximating nearby pocket-coordinate perturbations.

Proof. Apply a second-order Taylor expansion to $\ell(f_\theta(\cdot, \mathcal{P} + \boldsymbol{\eta}), t)$ around $\mathcal{P}$. The first-order term vanishes after expectation because $\mathbb{E}[\boldsymbol{\eta}] = 0$, and the second-order term contracts the pocket Hessian with $\mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top] = \sigma_p^2 I$, giving Eq. (41). For squared denoising loss, the Hessian decomposes into a Gauss–Newton term $J_{\mathcal{P}} f_\theta^\top J_{\mathcal{P}} f_\theta$ plus residual-weighted second-derivative terms; near a well-fit denoiser the residual terms are small, leaving the stated Jacobian penalty. □

This local analysis also explains why the perturbation can be made noise-level dependent. At large ligand noise σ, the denoiser should use coarse pocket information and can tolerate stronger pocket smoothing; at small σ, the reverse process refines local geometry and should not blur steric constraints. We therefore use a bounded schedule $\sigma_p = \min(0.1\sigma, 0.5)$ in the main setting: the proportional term keeps pocket uncertainty below the ligand corruption scale, while the cap prevents the augmentation from erasing Angstrom-scale pocket geometry. From an information-bottleneck perspective [41], this schedule suppresses nuisance information about the exact crystallographic pocket realization while preserving the pocket information needed to predict ligand geometry and affinity. If σ_p is too small, the Jacobian penalty is negligible and the model can overfit to brittle pocket details; if it is too large, the conditioning channel loses target-specific information and the model shifts toward a weaker, less precise conditional distribution. In this sense, pocket perturbation controls the local Lipschitz behavior of the pocket-conditioning channel: it smooths the denoiser's response to pocket geometry and only indirectly smooths the effective conditional landscape explored by the sampler. The ablation in Table 4 is consistent with this bias–variance trade-off: the mild bounded schedule improves local distance fidelity, atom-type JS, Vina metrics, and centroid distance, the corresponding unbounded 0.1σ schedule remains competitive but is slightly weaker on the same target-specific metrics, the lighter $\min(0.05\sigma, 0.25)$ schedule further reduces strain, and the stronger $\min(0.3\sigma, 1.5)$ schedule favors broader distributional and SA-oriented statistics but gives up some target-specific precision.

F Guidance Scale Sensitivity Details

Table 5 and Figures 6–7 provide the step-wise CFG-scale diagnostics corresponding to the main-text trade-off trajectory in Figure 2. They make the same trend explicit from metric-wise views: weak-to-moderate guidance improves affinity-related and molecular-property objectives while preserving strong geometry, whereas overly strong guidance eventually reduces $\text{Valid}_{3\text{D}}$ and increases strain and centroid error.

G RampUp Guidance Schedule

In addition to the constant CFG scale used in the main experiments, we also consider a simple RampUp variant inspired by recent analyses of guidance schedules in masked diffusion [36]. The motivation is that strong extrapolation at the beginning of sampling can be brittle, since the ligand state is still dominated by high noise and the conditional and dropped-condition predictions may

Table 4: Sensitivity to protein-side perturbation strategies. Local JSD averages the eight short-range pairwise distance JSDs over 6-6|1, 2, 4, 6-7|1, 2, 4, and 6-8|1, 2, corresponding to C–C, C–N, and C–O distance statistics. JSD-All-12Å pools all intraligand atom-pair distances up to 12 Å, JSD-CC-2Å focuses on short-range carbon–carbon distances, and Atom reports atom-type frequency JS. Lower JSD, Vina metrics, strain energy, and centroid distance are better; higher QED, SA, Valid_{3D}, and clash-free rate are better.

	JSD (↓)				Property / Affinity				Geometry			
Perturbation	Local	12Å	CC-2Å	Atom	QED (↑)	SA (↑)	Vina (↓)	Vina Min (↓)	Valid_{3D} (↑)	Strain (↓)	Clash-Free (↑)	Centroid (↓)
None	0.270	0.069	0.276	0.143	0.63	0.73	-7.78	-7.99	77.3	125.9	94.07	1.271
Fixed 0.1	0.262	0.064	0.274	0.140	**0.65**	0.72	-7.70	-8.15	80.0	129.2	93.15	1.257
0.1σ	0.267	0.068	0.269	0.140	0.64	0.74	-7.83	-8.07	**81.3**	122.5	94.27	1.307
$\min(0.05\sigma, 0.25)$	0.270	0.070	0.288	0.145	0.64	0.74	-7.86	-8.16	81.2	**121.0**	94.13	1.275
$\min(0.1\sigma, 0.5)$	**0.255**	0.065	0.269	**0.120**	0.64	0.71	**-7.89**	**-8.19**	80.6	127.9	94.02	**1.255**
$\min(0.2\sigma, 1.0)$	0.269	0.069	0.281	0.137	0.64	0.74	-7.77	-8.08	80.2	121.8	**94.28**	1.286
$\min(0.3\sigma, 1.5)$	0.257	**0.063**	**0.263**	0.132	0.64	**0.75**	-7.66	-7.98	79.4	118.5	93.67	1.280

Table 5: Sensitivity to CFG scale. We report median property metrics together with key GenBench3D geometry metrics.

s	QED (↑)	SA (↑)	Vina Score (↓)	Valid_{3D} (↑)	Strain (↓)	Clash-Free (↑)	Centroid Dist. (↓)
0	0.48	0.63	-6.70	78.64	164.9	90.59	1.25
0.5	0.57	0.66	-7.03	78.74	152.3	92.02	**1.24**
1	0.60	0.70	-7.20	77.99	127.4	90.74	1.25
2	0.62	0.71	-7.33	78.92	**125.6**	92.15	1.26
3	0.63	0.71	-7.56	**81.12**	127.7	93.57	1.25
5	**0.64**	0.71	-7.89	80.60	127.9	94.02	1.26
10	**0.64**	**0.72**	**-8.12**	78.54	137.7	**95.03**	1.28
15	**0.64**	0.70	-8.00	75.14	160.4	94.89	1.31
20	0.60	0.69	-7.61	72.04	187.5	94.51	1.38

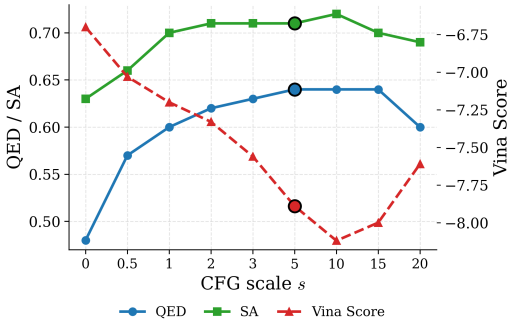


Figure 6: Property-side trends under different CFG scales. QED, SA, and Vina Score improve from weak to moderate guidance, with Vina Score reaching its strongest value around $s = 10$ and QED/SA saturating around the moderate-to-strong range.

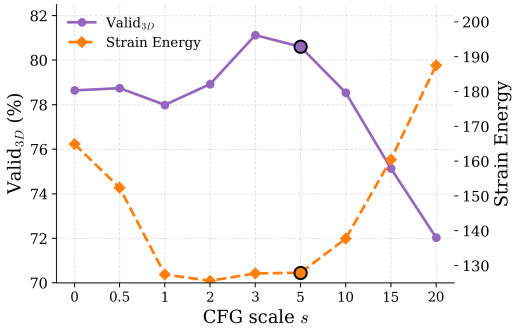


Figure 7: Geometry-side trends under different CFG scales. Moderate guidance keeps Valid_{3D} high and strain energy low, whereas overly strong guidance reduces Valid_{3D} and increases strain. In this study, $s = 5$ is used as the representative operating point.

be poorly calibrated. Instead of applying a fixed scale s at every reverse step, RampUp uses a time-dependent scale

$$s_i^{\text{ramp}} = \alpha_i s, \qquad \alpha_i = \frac{i}{N-1}, \qquad i = 0, \ldots, N-1, \tag{42}$$

where N is the number of sampling steps and i indexes the reverse trajectory from the highest-noise step to the final low-noise step. The guided coordinate and atom-type predictions then use the same CFG rule as Eq. (8), with s replaced by s_i^{ramp} at step i. Thus, early steps remain close to the property-unconditional ($s = 0$) trajectory, while later low-noise steps receive the full property-steering strength. This schedule does not require retraining or additional model evaluations; it only changes the scalar guidance multiplier used during inference.

We use RampUp as an auxiliary diagnostic rather than as the default sampler. Since the same nominal scale is not directly comparable across constant and RampUp schedules, we compare them at similar median docking strength. In the mid-guidance regime, RampUp gives a comparable trade-off: for example, constant CFG with $s = 2$ gives median Vina Score -7.33 and median Vina Min -7.78, while RampUp with final scale $s = 5$ gives -7.49 and -7.81. At this matched operating point, RampUp keeps a similar distributional profile (JSD-All 0.0488 versus 0.0490) and slightly higher completion rate (0.960 versus 0.954), with comparable molecular stability (34.7% versus 34.4%). At stronger docking targets, however, this advantage is not consistent; matching the constant $s = 5$ median Vina Score with RampUp requires a larger final scale and lowers molecular stability. We therefore treat RampUp as a conservative schedule that can improve the median trade-off in a moderate guidance range, rather than as a uniformly better replacement for constant CFG.

H Component-Wise Ablation Summary

To make the contribution of each design choice easier to inspect, we collect the main component-wise comparisons in Table 6. The table reorganizes results by design axis rather than by metric category. Several entries are repeated from Tables 1, 5, and 4 to make the attribution easier to read in one place. All PocketVE variants use the same test split and GenBench3D evaluation protocol, and guided variants use the same trained checkpoint unless the ablated component changes training. The schedule row reports an additional ablation, while the other rows reorganize results from the main tables by design axis.

Table 6: Component-wise ablation summary for PocketVE.

Ablation axis	Setting	QED	SA	Vina Score	Valid_{3D}	Strain	Clash-Free	Centroid
Backbone/guidance baseline	TAGMol (guided)	0.56	0.56	-7.77	58.6	457.4	93.21	1.54
VE backbone without CFG	PocketVE ($s = 0$)	0.48	0.63	-6.70	78.64	164.9	90.59	1.25
Plain conditional sampling	PocketVE ($s = 1$)	0.60	0.70	-7.20	77.99	127.4	90.74	1.25
Moderate CFG	PocketVE ($s = 5$)	0.64	0.71	-7.89	80.60	127.9	94.02	1.26
Protein perturbation	No perturbation	0.63	0.73	-7.78	77.3	125.9	94.07	1.271
Optimizer	Adam ($s = 5$)	0.64	0.73	-7.84	78.34	152.8	93.74	1.272
Optimizer	AdamW ($s = 5$)	0.63	0.73	-7.40	76.62	170.5	93.58	1.340
Noise schedule	Log-uniform schedule ($s = 5$)	0.64	0.70	-7.13	73.11	186.8	90.77	1.303
Sampler control	No sampling-time noise	0.59	0.63	-5.75	59.70	495.7	75.70	1.44

The comparison between TAGMol (guided) and PocketVE ($s = 0$) should be read as the bundled effect of switching to the PocketVE backbone and sampler without guidance, including the VE-style coordinate parameterization, sampling-time noise injection, and generalized arcsin schedule. Because the TAGMol reference itself uses guidance, this gap also includes the removal of TAGMol's own guidance mechanism. Together with the cumulative ablation in Table 2, this shows that the geometric stabilization mainly comes from these backbone-level sampling and parameterization changes before inference-time guidance is applied. Concretely, Valid_{3D} increases by about 20 points, while strain energy is reduced by about 64% relative to guided TAGMol. The comparison from $s = 0$ to $s = 5$ then isolates inference-time CFG under the same checkpoint and sampling procedure: moderate guidance improves Vina Score, QED, and SA while preserving high 3D validity. The no-perturbation row shows that pocket-side perturbation is not a uniformly dominant switch, but it changes the property–geometry frontier and provides a useful reference point in the current setting. The optimizer rows provide controls at the same CFG scale. Adam, which follows the optimizer choice used

in TAGMol, remains a strong baseline, while AdamW-only training is weaker in both affinity and geometry. The main Muon–AdamW setting still gives the best overall geometry and strain, suggesting that the final gains are not simply due to replacing Adam with AdamW. The log-uniform-schedule row further shows that the reported gains are not produced by CFG alone: replacing the generalized arcsin schedule with log-uniform sampling at the same guidance scale leads to weaker Vina Score, lower Valid_{3D}, higher strain, and worse centroid error.

I Repeated-Run Variability

To document the stability of the final operating point without forcing the reader to cross-reference the main table, Table 7 reports the final PocketVE metrics together with their repeated-run standard deviations under the same evaluation protocol. We keep this summary compact and focus on the main distribution, property, affinity, and geometry metrics used throughout the paper.

Table 7: Final PocketVE performance with repeated-run variability. Each entry is the median across three independent runs $\pm$ standard deviation; the Table 1 run is one of the three. We report the same representative statistics used in the main text: QED_{med}, SA_{med}, $\text{Vina Score}_{\text{med}}$, $\text{Vina Min}_{\text{med}}$, Valid_{3D}, $\text{Strain}_{\text{med}}$, Clash-Free, Centroid mean, Local, JSD-All-12Å, JSD-CC-2Å, and atom-type JS. Valid_{3D} and Clash-Free use the same displayed units as the main table.

QED ↑	SA ↑	Vina ↓	Vina Min ↓	Valid_{3D} ↑	Strain ↓	Clash-Free ↑	Centroid ↓
0.63 ± 0.0018	0.72 ± 0.0047	-7.734 ± 0.0150	-8.026 ± 0.0204	79.27 ± 0.68	124.7 ± 1.29	93.89 ± 5.91	1.284 ± 0.0127
Local ↓	12Å ↓	CC-2Å ↓	Atom ↓				
0.245 ± 0.0020	0.0629 ± 0.0003	0.265 ± 0.0004	0.121 ± 0.0003				

J Reference-Normalized Multi-Objective Hits

Table 8 reports an auxiliary reference-normalized diagnostic rather than a replacement for the main benchmark metrics. The goal is to test whether a method produces same-pocket samples that are jointly no worse than the reference ligand in docking affinity, QED, and SA. Among the filtered methods, PocketVE achieves the highest Joint-Ref average and Joint Success rate.

Table 8: Reference-normalized multi-objective hit metrics; all percentage metrics are higher-is-better. Overall QED, SA, and diversity averages are included to preserve the average property statistics omitted from the compact main table. High Affinity follows prior SBDD usage and denotes the per-pocket fraction of generated molecules whose Vina Dock score is no worse than the reference ligand, and Success is the percentage of pockets with at least one such molecule. High-Aff. Subset reports average QED and SA within the High Affinity subset. Hit Rate is the fixed-threshold multi-objective rate with QED ≥ 0.4, SA ≥ 0.5, and Vina Dock ≤ -8.18 kcal/mol. Joint-Ref is the per-pocket fraction of generated molecules that simultaneously satisfy Vina Dock, QED, and SA no worse than the reference ligand. Joint Success is the percentage of pockets with at least one Joint-Ref molecule.

Method	Hit Rate	Overall Avg.			High Affinity			High-Aff. Subset		Joint-Ref		
		QED	SA	Div.	Avg.	Med.	Success	QED Avg.	SA Avg.	Avg.	Med.	Success
AR	12.9	0.51	0.63	0.70	41.2	37.0	72.2	0.52	0.59	4.6	0.0	36.1
Pocket2Mol	24.3	0.56	**0.74**	0.69	48.0	51.0	88.0	0.57	**0.72**	16.5	8.0	74.0
TargetDiff	20.5	0.48	0.58	0.72	57.6	58.3	99.0	0.50	0.56	5.7	2.0	58.0
DecompDiff	27.7	0.45	0.61	0.68	63.8	75.0	91.9	0.44	0.59	6.8	1.1	50.5
IPDiff	27.4	0.52	0.61	**0.74**	68.2	74.6	98.0	0.52	0.57	10.0	2.6	70.0
TAGMol (guided)	27.7	0.55	0.56	0.69	68.6	75.7	99.0	0.55	0.54	7.1	1.5	58.0
BindDM	24.9	0.51	0.58	-	64.2	66.1	**100.0**	0.53	0.55	6.9	1.2	67.0
ALiDiff	25.2	0.50	0.57	0.73	68.5	76.0	**100.0**	0.52	0.54	7.0	0.0	48.0
SeFMol	37.4	**0.63**	0.60	0.69	68.7	76.3	**100.0**	**0.63**	0.57	11.2	4.0	68.0
PAFlow	31.5	0.49	0.57	0.71	**80.8**	**93.7**	99.0	0.50	0.56	8.9	1.1	52.0
PocketVE	**47.3**	0.61	0.73	0.72	73.9	85.2	**100.0**	0.62	0.70	**26.4**	**20.9**	**92.0**

K Additional Property–Validity Trade-Offs

Figure 8 extends the main binding–geometry analysis to the drug-likeness and synthesizability axes. The same qualitative trade-off remains visible: weak-to-moderate guidance improves the property side without immediately leaving the high-validity regime, whereas stronger guidance eventually pushes the model toward lower Valid_{3D}. We place these plots in the appendix because they reinforce the same guidance-scale story as Figure 2, but from auxiliary property views rather than the primary affinity view.

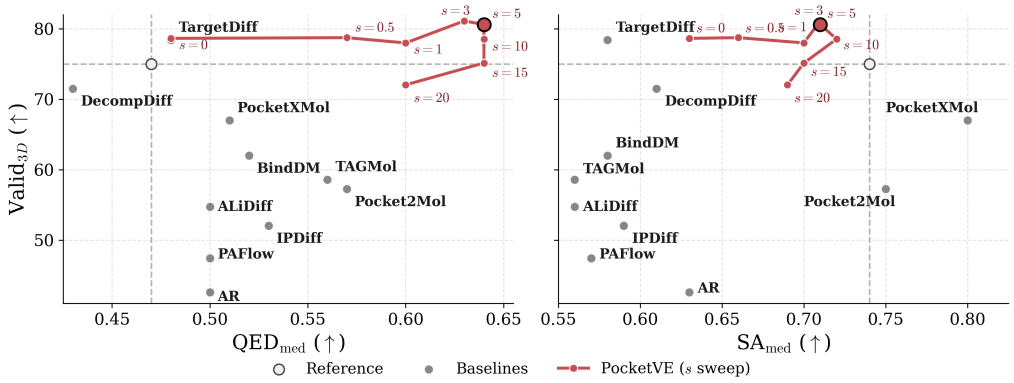


Figure 8: Property–validity trade-off across baselines and PocketVE guidance scales. The left and right panels report QED–Valid_{3D} and SA–Valid_{3D}, respectively. Only baselines with available GenBench3D Valid_{3D} values are shown. PocketVE moves toward higher drug-likeness and synthesizability while retaining high 3D validity under moderate CFG scales.

L Additional Distance-Distribution Diagnostics

Figure 9 makes the JSD diagnostics in the main text visually inspectable rather than purely numeric. Its purpose is to show what the all-atom distance divergence actually looks like in distribution space, and to verify that the quantitative JSD ranking in Figure 3 corresponds to visible shifts in the intraligand distance profile. Methods with larger JSD show broader distortions in the short- and mid-range distance modes, whereas PocketVE remains comparatively close to the reference distribution.

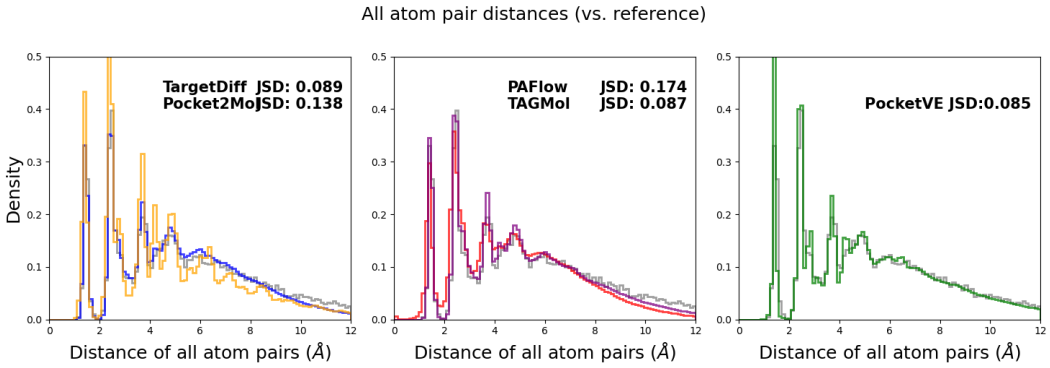


Figure 9: All-atom pair distance distributions compared against the reference ligands in the test set, following the same TargetDiff-style logic used for JSD diagnostics in the main text. Each panel overlays the empirical distribution of reference intraligand atom-pair distances with the corresponding generated distribution, and reports the resulting Jensen–Shannon divergence. The PocketVE panel corresponds to guided sampling at $s = 5$. Lower JSD indicates closer agreement with the reference distance profile, but this remains a distribution-fidelity diagnostic rather than an explicit physical-validity metric.

M Test-Time Protein Perturbation

We use this experiment as a controlled stress test rather than as a standard evaluation setting. At inference time, Gaussian noise is added only to the protein pocket coordinates, while ligand initialization, sampling schedule, and CFG scale are kept fixed at the main operating point $s = 5$. This setup probes whether training-time pocket perturbation improves tolerance to small errors in the conditioning geometry.

Table 9: Test-time protein perturbation stress test under inference-time pocket noise. Gaussian noise with scale σ_{pocket} is added to pocket coordinates at inference time. We compare PocketVE trained without protein perturbation, PocketVE trained with $\min(0.1\sigma, 0.5)$ perturbation, and PocketVE trained with $\min(0.3\sigma, 1.5)$ perturbation using the same four JSD diagnostics as in the main text together with median property metrics and key GenBench3D geometry metrics. σ_{pocket} is measured in Å. Lower Local, 12Å, CC-2Å, Atom, Vina, Vina Min, Strain, and Centroid are better; higher QED, SA, Valid_{3D}, and Clash-Free are better.

		JSD (↓)				Property / Affinity				Geometry			
Method	σ_{p}	Local	12Å	CC-2Å	Atom	QED ↑	SA ↑	Vina ↓	Vina Min ↓	Valid_{3D} ↑	Strain ↓	Clash-Free ↑	Centroid ↓
PocketVE w/o perturb.	0.0	0.265	0.0690	0.2750	0.1422	0.63	0.73	-7.83	-8.12	76.17	121.7	94.61	1.274
	0.1	0.264	0.0672	0.2739	0.1419	0.65	0.73	-7.81	-8.15	79.48	127.7	94.76	1.301
	0.3	0.269	0.0678	0.2754	0.1487	0.63	0.73	-7.10	-7.95	78.03	136.2	90.34	1.337
	0.5	0.270	0.0713	0.2832	0.1552	0.64	0.74	-5.90	-7.58	79.28	130.6	81.14	1.403
	1.0	0.275	0.0693	0.2802	0.1655	0.61	0.72	-2.90	-6.60	76.69	159.7	44.80	1.795
PocketVE w/ $\min(0.1\sigma, 0.5)$	0.0	0.255	0.0646	0.2674	0.1225	0.65	0.71	-7.87	-8.18	80.05	128.4	93.79	1.250
	0.1	0.263	0.0645	0.2689	0.1235	0.63	0.72	-7.65	-7.98	79.72	122.3	94.33	1.283
	0.3	0.261	0.0640	0.2649	0.1261	0.64	0.73	-6.82	-7.65	78.43	117.6	91.26	1.281
	0.5	0.268	0.0660	0.2745	0.1297	0.64	0.72	-5.68	-7.38	80.12	124.2	77.90	1.349
	1.0	0.263	0.0629	0.2660	0.1293	0.63	0.70	-2.75	-6.63	80.56	149.1	41.26	1.651
PocketVE w/ $\min(0.3\sigma, 1.5)$	0.0	0.265	0.0639	0.2637	0.1305	0.63	0.75	-7.64	-7.98	78.96	115.8	92.93	1.273
	0.1	0.261	0.0652	0.2613	0.1340	0.65	0.76	-7.48	-7.87	80.68	104.7	94.32	1.294
	0.3	0.266	0.0671	0.2664	0.1341	0.65	0.76	-6.73	-7.53	80.77	110.4	89.63	1.316
	0.5	0.265	0.0653	0.2646	0.1393	0.65	0.77	-5.78	-7.36	81.15	110.5	78.95	1.353
	1.0	0.266	0.0672	0.2658	0.1441	0.62	0.74	-2.52	-6.30	76.45	134.6	41.93	1.589

Across the sweep, both perturbation-trained variants generally preserve lower distance-distribution JSDs than the model trained without pocket perturbation. The stronger 0.3σ training perturbation further improves SA and strain-energy behavior at several noise levels, while the milder $\min(0.1\sigma, 0.5)$ training perturbation retains the strongest Vina-based scores at low test-time noise. However, severe test-time corruption still substantially weakens Vina-based scores and clash-free rates for all three models. These results suggest that protein perturbation mainly acts as a robustness regularizer for the conditioning geometry, rather than making the sampler invariant to arbitrarily corrupted pockets.

N Pocket-Diagnostic Details

For pocket shuffling, we apply a fixed cyclic permutation with no self-matches across the 100 test pockets. Each generated ligand is translated from its source pocket center to the target pocket center, then evaluated against the mismatched target. Correct and shuffled conditions contain the same 9,471 ligands. We first aggregate within each target pocket, average pockets equally, and form percentile 95% CIs from 10,000 paired pocket-bootstrap replicates. This is a post-hoc full-pipeline specificity control; it does not regenerate ligands while conditioning on an incorrect pocket and does not isolate a causal network component.

For PoseCheck, we evaluate original poses without redocking, summarize poses within each pocket, and average the 100 pockets equally. Table 10 summarizes clashes, clash-free rates, and total interactions, and Table 11 gives the full contact-category profile. The changes are mixed: PocketVE does not maximize every interaction category, and the reference ligands themselves average 11.79 total interactions, close to PocketVE's 11.99. Thus, total contact count should not be interpreted as interaction quality.

Table 10: Pocket-equal PoseCheck estimates on original poses with 10,000-replicate pocket-bootstrap 95% CIs. PoseCheck clash-free denotes the percentage of poses with zero PoseCheck clashes and differs from the GenBench3D clash-free metric in Table 1.

Method	Mean clashes ↓	Clash-free (%) ↑	Mean interactions
PocketVE	**8.52 [7.12, 10.10]**	**4.54 [2.91, 6.60]**	11.99 [10.98, 13.01]
TargetDiff	12.95 [11.19, 14.89]	1.65 [1.04, 2.33]	13.14 [12.15, 14.15]
PAFlow	10.36 [7.87, 14.14]	3.19 [1.88, 4.73]	11.65 [10.78, 12.53]
TAGMol	13.88 [11.81, 16.17]	0.91 [0.31, 1.61]	12.76 [11.83, 13.72]
Reference	7.79 [6.49, 9.14]	5.00 [1.00, 10.00]	11.79 [10.63, 12.99]

Table 11: Pocket-equal PoseCheck contact-category estimates with 10,000-replicate pocket-bootstrap 95% CIs.

Method	Total interactions	HBD	HBA	Hydrophobic	VdW	Mean clashes
PocketVE	11.99 [10.98, 13.01]	0.36 [0.30, 0.42]	1.41 [1.14, 1.68]	1.06 [0.88, 1.25]	9.16 [8.47, 9.89]	8.52 [7.12, 10.10]
TargetDiff	13.14 [12.15, 14.15]	0.68 [0.60, 0.76]	1.63 [1.37, 1.90]	1.15 [0.97, 1.35]	9.69 [9.02, 10.38]	12.95 [11.19, 14.89]
PAFlow	11.65 [10.78, 12.53]	0.39 [0.33, 0.45]	1.23 [1.00, 1.47]	1.59 [1.36, 1.83]	8.44 [7.86, 9.05]	10.36 [7.87, 14.14]
TAGMol	12.76 [11.83, 13.72]	0.59 [0.51, 0.68]	1.50 [1.26, 1.76]	1.17 [0.97, 1.38]	9.50 [8.87, 10.15]	13.88 [11.81, 16.17]
Reference	11.79 [10.63, 12.99]	0.50 [0.36, 0.65]	1.96 [1.57, 2.37]	0.72 [0.49, 0.98]	8.61 [7.86, 9.38]	7.79 [6.49, 9.14]

Table 12 reports paired differences with the sign convention PocketVE minus comparator. Negative clash differences and positive clash-free differences favor PocketVE; interaction-count differences are descriptive because larger is not uniformly better.

Table 12: Paired pocket-level PoseCheck differences with percentile 95% CIs. Clash-free differences are percentage points.

Comparator	Δ mean clashes	Δ clash-free	Δ total interactions
TargetDiff	-4.42 [-5.92, -3.28]	+2.90 [+1.61, +4.64]	-1.15 [-1.47, -0.85]
PAFlow	-1.84 [-5.47, +0.63]	+1.35 [-0.30, +3.40]	+0.34 [-0.18, +0.88]
TAGMol	-5.36 [-7.26, -3.78]	+3.63 [+2.05, +5.67]	-0.77 [-1.25, -0.24]
Reference	+0.73 [-0.60, +2.17]	-0.46 [-4.37, +2.79]	+0.20 [-0.34, +0.73]

O Property-Steering Diagnostics

We evaluate non-default and conflicting condition vectors with the same fixed PocketVE checkpoint, $s = 5$, 100 pockets, and 10 samples per pocket. The condition order is (Vina, QED, SA), and larger indices denote more favorable training bins. Table 13 shows that ordered requests move all three aggregate medians in the requested directions. Conflicting requests also expose cross-property competition: for example, $(4, 0, 2)$ improves Vina but lowers QED relative to $(2, 4, 2)$, while $(2, 0, 4)$ and $(2, 4, 0)$ reverse the QED–SA preference at similar Vina levels. The non-monotonic outcomes preclude an interpretation as calibrated independent control.

Table 14 reports distribution-level attainment under favorable joint requests. Rates use successfully evaluated molecules as the denominator. The Vina column uses `vina_score`, whereas training boundaries use `vina_dock`; consequently, we report the score shift but do not treat it as strict Vina-bin attainment.

P CFG Diversity and Reference Coverage

Table 15 aggregates successfully evaluated molecules across all pockets. Higher CFG mildly reduces molecule-level uniqueness and increases fingerprint similarity, but scaffold entropy and top-scaffold concentration do not deteriorate monotonically. Reference-scaffold coverage decreases with guidance. Because these summaries pool molecules across pockets, they characterize global chemical-space behavior and can mask within-pocket convergence or pocket-specific reference recovery.

Table 13: Aggregate property medians under ordered and conflicting requested bins. All rows use the same checkpoint and $s = 5$.

Requested bins (Vina, QED, SA)	Vina Score median	QED median	SA median
$(0, 0, 0)$	-3.294	0.160	0.520
$(1, 1, 1)$	-5.177	0.284	0.540
$(2, 2, 2)$	-7.099	0.589	0.660
$(3, 3, 3)$	-7.288	0.619	0.690
$(4, 4, 4)$	-7.881	0.642	0.710
$(4, 0, 2)$	-7.515	0.604	0.700
$(2, 4, 2)$	-7.144	0.670	0.650
$(2, 0, 4)$	-6.937	0.392	0.690
$(2, 4, 0)$	-6.865	0.605	0.530

Table 14: Property-distribution diagnostics under CFG. QED+SA top requires both favorable property bins; all three top additionally reports the more stringent three-way criterion using the available score-only Vina boundary. Vina values are interpreted directionally because the evaluation and training metric variants differ.

Scale	Evaluated n	QED top	SA top	QED+SA top	All three top	Vina median
0	935	14.01%	7.06%	1.28%	0.00%	-6.704
1	961	22.58%	12.59%	3.02%	0.00%	-7.196
5	953	27.91%	19.31%	3.99%	0.10%	-7.892
10	925	**30.59%**	**21.51%**	**5.51%**	**0.11%**	**-8.124**

Q Coordinate-Scale Robustness and Diagnostics

We evaluate $\sigma_{\text{data}} \in \{5, 7.5, 10, 12.5, 15\}$ on the same 100 test pockets with 100 generated ligands per pocket, CFG scale $s = 5$, and the generalized-arcsin scheduler. Accordingly, Table 16 is a robustness sensitivity analysis over the evaluated scale range rather than a sharply optimized ranking.

No setting dominates all metrics: $\sigma_{\text{data}} = 10$ has the highest evaluation success, 5 has the most favorable mean Vina score, 7.5 has the highest SA and lowest median strain, and 15 has the lowest evaluation success. Distributional metrics are similarly stable and non-monotonic: JSD-All-12Å ranges from 0.0592 to 0.0665, JSD-CC-2Å from 0.2533 to 0.2819, and atom-type JS from 0.1215 to 0.1458. We therefore use $\sigma_{\text{data}} = 10$ as a fixed operating choice within a robust range, not as a sharply optimized constant.

Table 17 complements the scale sweep with targeted backbone and coordinate-scale diagnostics. The sampler-control result is reported with the component-wise ablation in Appendix H; removing sampling-time noise sharply lowers evaluation success and geometric quality, indicating that the reverse-process noise treatment is an important part of the integrated sampler. The VEDA-style row retains the TAGMol architecture but uses $\sigma_{\text{data}} = 1$ and $s = 0$, so it tests the bundled formulation rather than isolating σ_{data}. Removing the σ_{data} factor in Eq. (32) leaves reconstruction and ligand-only geometry metrics similar but destroys pocket-aware spatial compatibility, as reflected by pathological Vina scores and a collapse in clash-free poses. The no-Eq. (32) row is interpreted as a targeted compatibility diagnostic rather than as an isolated causal estimate of all geometric gains.

Table 15: Global diversity and reference-coverage diagnostics under CFG. Reference coverage is the percentage of reference scaffolds recovered in the generated set.

	Uniqueness		Scaffold			Similarity / Coverage	
Scale	Molecule	Scaffold	Entropy	Top-1	Top-10	Pairwise Tan.	Ref. Coverage
0	**99.36%**	83.42%	0.9481	6.95%	15.19%	**0.0919**	**13.70%**
1	98.65%	86.16%	0.9616	**5.31%**	12.59%	0.1072	12.33%
5	97.27%	**86.78%**	**0.9636**	5.35%	**12.17%**	0.1176	9.59%
10	96.97%	84.22%	0.9563	6.38%	13.41%	0.1184	8.22%

Table 16: σ_{data} sweep (100 pockets $\times$ 100 ligands). Brackets are pocket-bootstrap 95% CIs for metrics recoverable per pocket; QED, SA, and Vina are evaluator-level point estimates.

σ_{data}	Eval. success [CI]	QED	SA	Vina Score	Valid$_{3D}$ [CI]	Strain median [CI]	Clash-free [CI]
5.0	93.79 [92.11, 95.25]	0.614	0.736	-7.453	79.01 [77.31, 80.73]	126.46 [111.74, 137.27]	94.47 [90.63, 97.62]
7.5	93.33 [91.72, 94.78]	0.606	0.764	-7.216	80.65 [78.91, 82.36]	102.83 [91.59, 113.09]	93.72 [89.65, 96.96]
10.0	94.55 [93.24, 95.68]	0.613	0.725	-7.370	79.48 [77.75, 81.12]	126.16 [113.83, 139.25]	94.01 [90.05, 97.26]
12.5	93.47 [92.03, 94.74]	0.608	0.736	-7.430	78.66 [77.01, 80.35]	128.86 [114.42, 143.01]	94.23 [90.20, 97.37]
15.0	91.85 [90.42, 93.21]	0.597	0.748	-7.306	81.73 [80.13, 83.27]	125.58 [112.58, 139.49]	93.70 [89.40, 97.10]

Table 17: Backbone and coordinate-scale diagnostics (100 pockets $\times$ 10 ligands). QED, SA, and Vina are means; strain is the pooled median. The VEDA-style and no-Eq. (32) rows are targeted bundled diagnostics rather than isolated causal comparisons.

Setting	Eval. success	QED	SA	Vina	Valid$_{3D}$	Strain	Clash-free	Centroid (Å)
PocketVE setup	95.30	0.624	0.722	-7.589	78.72	127.94	94.23	1.25
VEDA-style ($\sigma_{\text{data}} = 1, s = 0$)	89.50	0.400	0.563	-5.454	43.43	1007.53	91.40	1.19
Without scale preservation	97.50	0.628	0.785	+92.237	82.39	100.41	3.39	3.33

R Additional Case Studies

Figure 10 provides three further case studies. It compares the reference pose with representative baseline generations and the PocketVE sample for specific binding pockets. These examples are included to show that the qualitative trends discussed in the main text are not limited to a single target.

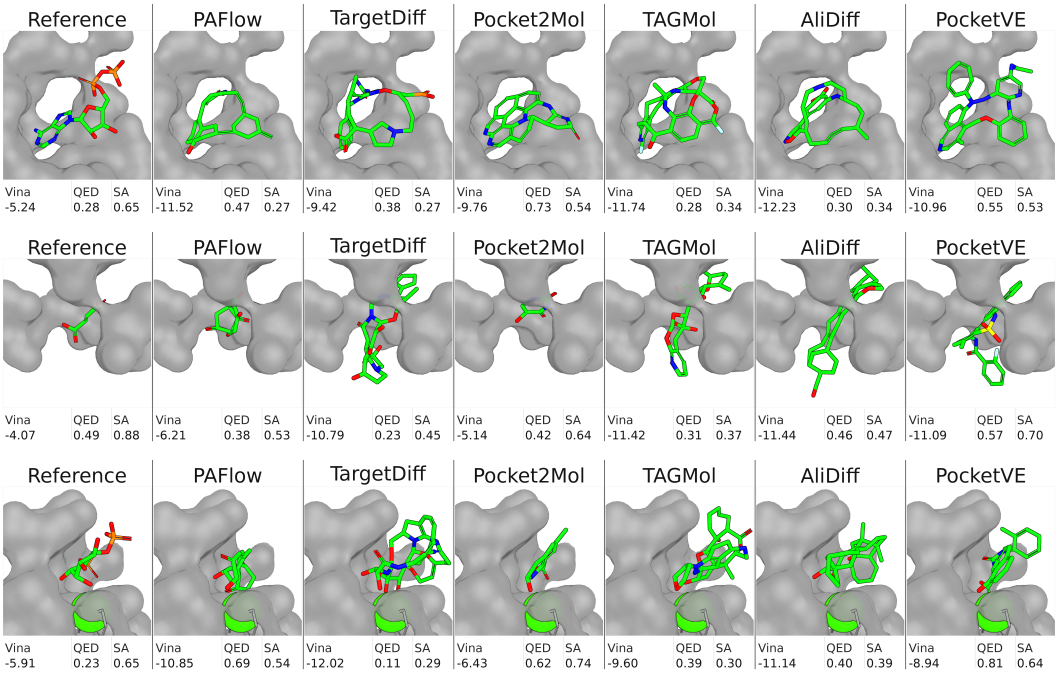


Figure 10: Additional case studies for PDB ID 5W2G (top), 3W83 (middle), and 1DJY (bottom).

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 abstract and introduction characterize PocketVE as an integrated pocket-conditioned formulation with a favorable geometry–property trade-off and coarse inference-time steering on the filtered CrossDocked2020 split. They do not attribute the gains to VE alone or claim calibrated control, uniform metric dominance, experimental binding, or broad protein-family generalization.

   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: Section 5 discusses inference-time cost, target-dependent guidance, simplified pocket perturbation, the benchmark scope, and limits on component attribution and docking-based evaluation. Appendices N, O, and P explain how to interpret the pocket diagnostics, property steering, and pooled diversity results.

   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 a formal theoretical statement together with its proof for the regularization effect of pocket perturbation. The proposition, its assumptions, and the complete proof are provided in Section E.

   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 model formulation, sampling procedure, training setup, dataset preprocessing, evaluation protocol, and main hyperparameter choices, and we also provide code and data in the supplemental material. Together, these materials are intended to make the main experimental results reproducible.

   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 data and code in the supplemental material together with instructions for reproducing the main experimental results. These materials are intended to allow reviewers to faithfully reproduce our reported results under the same setup.

   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 describes the dataset filtering, evaluation protocol, metrics, sampling schedule, guidance settings, and optimizer choices needed to understand the reported results. These details are provided in Section 3.3, Section 3.4, Section 3.5, and Section 4.1.

   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: Section 4.2 reports paired pocket-bootstrap confidence intervals for the main comparisons. Appendices N and Q report bootstrap intervals for pocket shuffling, PoseCheck, and the coordinate-scale sweep. Formal significance conclusions are restricted to source-aligned paired comparisons.

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: Appendix C reports the training hardware and duration. Table 2 reports per-method sampling times and the runtime setup: one NVIDIA A100 GPU, batch size 10, and no mixed precision.

   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: The work uses public benchmark data and standard computational evaluation and does not involve human subjects. We are not aware of any aspect of the research that conflicts with the NeurIPS Code of Ethics.

   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: Section 1 motivates the work as a tool for early-stage drug design. Section 5 discusses the risk of overinterpreting docking scores as evidence of binding and identifies experimental validation as future work.

    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 does not release a high-risk general-purpose generative model or scraped dataset of the kind targeted by this question. The study focuses on protein-conditioned 3D molecular generation and does not introduce a release setting that requires special misuse safeguards in the current submission.

    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: We cite the original papers and public repositories for the external assets used in this work and follow their stated license terms. For example, for TargetDiff we cite both the original paper and the official repository (`https://github.com/guanjq/targetdiff`), whose code is released under the MIT License.

    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: [N/A]

    Justification: The paper does not introduce a new public dataset, benchmark, or standalone released model asset. Its main contribution is a new method, and the provided supplementary materials are intended to support reproducibility rather than to document a separate newly released asset.

    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 compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or 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 human subjects or crowdsourcing, so IRB approval or an equivalent review was not required for this study.

    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 in this paper does not use LLMs as an important, original, or non-standard methodological component.

    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.