Structure-based drug design by denoising voxel grids

Pedro O. Pinheiro [1] **Arian Jamasb** [1] **Omar Mahmood** [1] **Vishnu Sresht** [1] **Saeed Saremi** [1]

Abstract

We present VoxBind, a new score-based generative model for 3D molecules conditioned on protein structures. Our approach represents molecules as 3D atomic density grids and leverages a 3D voxel-denoising network for learning and generation. We extend the neural empirical Bayes formalism (Saremi & Hyvärinen, 2019) to the conditional setting and generate structure-conditioned molecules with a two-step procedure: (i) sample noisy molecules from the Gaussian-smoothed conditional distribution with underdamped Langevin MCMC using the learned score function and (ii) estimate clean molecules from the noisy samples with single-step denoising. Compared to the current state of the art, our model is simpler to train, significantly faster to sample from, and achieves better results on extensive *in silico* benchmarks—the generated molecules are more diverse, exhibit fewer steric clashes, and bind with higher affinity to protein pockets.

1. Introduction

The goal of *structure-based drug design* (SBDD) (Blundell, 1996; Anderson, 2003) is to generate molecules that bind with high affinity to specific 3D structures of target biomolecules. Traditional computational approaches, such as virtual screening, search over a library of molecules and score them to identify the best binders for a given target of interest (Lyu et al., 2019). However, random search is arguably very inefficient as the chemical space grows exponentially with molecular size (Fink et al., 2005).

Recently, many generative modeling methods have been proposed as alternatives to search-based SBDD (see Thomas et al. (2023) for a review). In this setting, the goal is to develop data-driven approaches that generate molecules (*i.e.*, *ligands*) *conditioned* on 3D protein binding sites (*i.e.*, *pockets*). Generative models have the promise of exploring the chemical space more efficiently and effectively than search-based approaches.

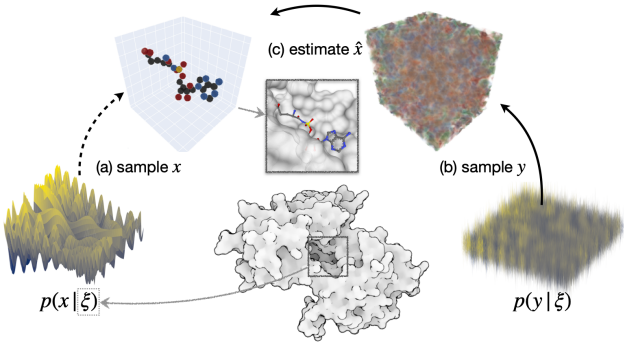

Figure 1. We are interested in sampling from $p(x|\xi)$, the distribution of ligands given pocket ξ. This is challenging due to the high-dimensionality of the data. Therefore, instead of (a) sampling directly from this distribution, we generate ligands in a two-step procedure: (b) sample y from the Gaussian-smoothed distribution $p(y|\xi)$ and (c) estimate the ligand $\hat{x}$ from y and ξ.

SBDD generative models typically represent molecules as discrete voxel grids or atomic point clouds. Voxel-based approaches (Wang et al., 2022a; Ragoza et al., 2022; Wang et al., 2022b) represent atoms (or electron densities) as continuous densities and molecules as a discretization of 3D space into voxel grids (a voxel is a discrete unit of volume). Point-cloud based approaches (*e.g.*, Luo et al. (2021); Schneuing et al. (2022)) treat atoms as points in 3D Euclidean space and rely on graph neural network (GNN) architectures. Current state of the art methods in data-driven SBDD (Guan et al., 2023a;b) operate on point clouds and are based on E(3) equivariant diffusion models (Hoogeboom et al., 2022) conditioned on protein pockets: they sample points from a Gaussian prior and iteratively apply the learned reverse conditional diffusion process (over continuous coordinates and discrete atom types and bonds) to generate molecules.

There is a clear trade-off between the two data representation choices. On one hand, GNNs can leverage SE(3)-equivariance inductive bias more easily than architectures that operate on voxels (Geiger & Smidt, 2022; Weiler et al., 2018). On the other hand, they are known to be less expres-

[1]Prescient Design, Genentech. Correspondence to: Pedro O. Pinheiro <pedro@opinheiro.com>, Saeed Saremi <saremis@gene.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

sive due to the message passing formalism (Xu et al., 2019; Morris et al., 2019; Pozdnyakov & Ceriotti, 2022). The expressivity of these models can be improved with higher-order message passing schemes (Batatia et al., 2022). However, they require additional computational cost and have not been applied to 3D generative models yet. Recently, it has been shown empirically that non-equivariant—but more expressive—models are competitive with equivariant models on different domains, from computer vision (Bai et al., 2021; Gruver et al., 2022) to molecule generation (Flam-Shepherd & Aspuru-Guzik, 2023; Pinheiro et al., 2023; Wang et al., 2023). In fact, equivariance can be learned from large data and strong data augmentations. Inspired by these findings, we propose a model for SBDD that prioritizes expressivity over SE(3)-equivariance inductive bias.

Our model, *VoxBind*, is a new voxel-based method for 3D ligand generation conditioned on pocket structures. The proposed model generates molecules by extending the *neural empirical Bayes* (NEB) framework (Saremi & Hyvärinen, 2019; Pinheiro et al., 2023) to the structure-conditional setting. Given a protein pocket ξ, instead of sampling ligands x directly from $p(x|\xi)$, we follow a two-step procedure: (i) sample noisy molecules y from the Gaussian-smoothed distribution $p(y|\xi)$ and (ii) estimate the clean ligand from y and ξ. Fig. 1 illustrates our approach. Sampling is done with Langevin Markov chain Monte Carlo (MCMC), which is known to work much better on the smoother distribution than on the original (Saremi & Hyvärinen, 2019). We train a conditional denoiser—a model that predicts a clean ligand, given a noisy version of it and its binding pocket—to approximate the conditional score function of the smoothed distribution and the ligand estimator, necessary for steps (i) and (ii), respectively.

VoxBind is fundamentally different than current state of the art, *i.e.*, diffusion models on point clouds. First, voxelized representations allow us to use the same type of denoising architectures used in score-based generative models on images. These neural networks are very flexible, scalable and work well in many conditional generation applications (Rombach et al., 2022; Saharia et al., 2022a;b). Second, convolutional filters (and self-attention layers applied on regular grids) can arguably capture 3D patterns, surfaces and shape complementarity—useful features for structure conditioning—more effectively than message passing (see the many-body representation hypothesis discussed by Townshend et al. (2020)). Third, contrary to diffusion models, the noise process used in our approach does not displace atoms in space. This provides a natural way to avoid clashes between generated ligands and pockets. Finally, our approach only requires a single (fixed) noise level, making training and sampling considerably simpler.

These differences are reflected on empirical results: VoxBind generates better ligands (conditioned on protein binding sites) on CrossDocked2020 (Francoeur et al., 2020), a standard dataset for this task. The molecules generated by our model bind with higher affinity to protein pockets (computed with standard docking score software), are more diverse, exhibit fewer steric clashes and lower strain energies.

Our contributions can be summarized as follows. We propose VoxBind: a new score-based generative model for structure-based drug design. The proposed method is new, simple and effective. We show that VoxBind outperforms current state of the art on an extensive number of metrics, while being significantly faster to generate samples.

2. Related work

2.1. Unconditional 3D molecule generation

Many recent 3D molecule generation methods represent atoms as points (with 3D coordinates, atom types and possibly other features) and molecules as a set of points. For instance, Gebauer et al. (2018; 2019); Luo & Ji (2022) propose autoregressive approaches to generation, where points (atoms) are sampled iteratively over time, while Köhler et al. (2020); Satorras et al. (2021) generate molecules using normalizing flows (Rezende & Mohamed, 2015). Hoogeboom et al. (2022) propose the equivariant diffusion model (EDM), a diffusion-based (Sohl-Dickstein et al., 2015) generative model that operates on point clouds. EDMs learn to denoise a diffusion process (operating on both continuous and categorical data) and generate molecules by iteratively applying the denoising network on an initial noise. Many follow-up work propose extensions to EDM (Huang et al., 2022; Vignac et al., 2023; Xu et al., 2023).

Ragoza et al. (2020) propose a different way to generate molecules: map atomic densities to discrete voxel grids and leverage computer vision techniques for generation. In particular, they propose a generation method based on variational autoencoders (Kingma & Welling, 2014). More recently, Pinheiro et al. (2023) propose VoxMol, a score-based generative model that operates on voxelized grids. They show that voxel-based representations achieve results comparable to point-cloud representations on unconditional molecule generation. Our work can be seen as an extension of VoxMol (and NEB) to the conditional generation setting.

2.2. Pocket-conditional molecule generation

The first methods proposed for this task represent molecules as 3D voxel grids and use 3D convnet architectures. For instance, Skalic et al. (2019) and Wang et al. (2022b) train models to generate ligand SMILES from voxelized pocket structures. Ragoza et al. (2022) uses conditional variational autoencoder to generate 3D ligands conditioned on vox-

elized pocket structures. Long et al. (2022) encodes molecular shapes with voxels and propose a model that generates molecular graphs conditioned on the voxelized shapes. Our method also utilizes voxelized molecules, but differs from previous work in terms of how we voxelize molecules, the network architecture and the generative model we use.

More recent methods, however, use point-cloud representations and SE(3)-equivariant graph neural networks. Luo & Ji (2022); Liu et al. (2022); Peng et al. (2022) propose conditional autoregressive methods that add atoms (and their corresponding bonds) one at a time, while Rozenberg et al. (2023) propose a method based on normalizing flows. Other authors propose methods that generate molecules by iteratively adding fragments conditioned either on ligand (Adams & Coley, 2022) or pocket (Zhang et al., 2023; Powers et al., 2023) shapes.

Current state of the art models for SBDD are based on point-cloud diffusion models. DiffSBDD (Schneuing et al., 2022) and TargetDiff (Guan et al., 2023a) adapt EDMs to the pocket-conditional generation setting. DecompDiff (Guan et al., 2023b) extends TargetDiff with three modifications: they decompose ligand into fragment priors, apply diffusion process on bonds (in addition to atom types and coordinates), and add guidance during generation process.

This work shows that we don't necessarily need point clouds or SE(3)-equivariant networks to achieve competitive results on structure-based drug design tasks. As we demonstrate in this paper, voxel-based representations—when coupled with a powerful generative model and an expressive network—can achieve state-of-the-art results.

2.3. Conditional image generation

Conditional image generation has been extremely successful and widely applied to many different computer vision tasks: text-to-image generation (Rombach et al., 2022), super-resolution (Saharia et al., 2022c), optical flow and depth estimation (Saxena et al., 2023), colorization (Saharia et al., 2022a), in-painting (Lugmayr et al., 2022) and semantic segmentation (Amit et al., 2021). All these methods share a commonality that inspired our approach: they adapt score-based generative models to the conditional setting by modifying the very expressive U-Net architecture (Ronneberger et al., 2015) to incorporate the conditioning signal.

3. Method

3.1. Voxelized molecules

We represent atoms as spherical densities in 3D space decaying exponentially with the square distance to their atomic center (see (10) in the appendix). Voxelized molecules are created by discretizing the space around atoms into voxel grids, where the value at each voxel represents atomic occupancy. Voxels take values between 0 (far away from all atoms) and 1 (at the center of atoms). Ligands and pockets are each represented as cubic grids with side length $L \in \mathbb{N}$. Each ligand grid and its corresponding pocket grid are centered around the center of mass of the ligand. We assume ligands have c_x atom types and pockets have c_ξ. Each atom type (element) is represented by a different grid channel (similar to R,G,B channels of images).

We consider a dataset with n voxelized ligand-pocket binding pairs $\{(x, \xi)_i\}_{i=1}^n$, where $x \in \mathbb{R}^{d_x}$ is the voxelized ligand and $\xi \in \mathbb{R}^{d_\xi}$ the voxelized pocket with dimensions $d_x = c_x L^3$ and $d_\xi = c_\xi L^3$, respectively. Following previous work (Pinheiro et al., 2023), we consider a fixed grid resolution of .25Å and a grid length $L = 64$. Therefore, the grids in the dataset occupy a volume of 16^3 cubic Ångströms. See Sec. A.1 (appendix) for further details on the voxelization procedure.

3.2. Conditional neural empirical Bayes

Let X and Ξ be the random variables associated with voxelized ligands and pockets, taking values in $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_\xi}$, respectively, with the joint density $p(x, \xi)$. Given $\Xi = \xi$, we would like to learn the smoothed conditional score function $\nabla_y \log p(y|\xi)$, where the smoothed conditional density $p(y|\xi)$ is obtained by convolving $p(x|\xi)$ with an isotropic Gaussian kernel:

$$p(y|\xi) = \int p(y|x)p(x|\xi)dx, \tag{1}$$

where

$$p(y|x) = \frac{1}{(2\pi\sigma^2)^{d_x/2}} \exp\Big(-\frac{\|y - x\|^2}{2\sigma^2}\Big), \tag{2}$$

equivalently $Y|x \sim \mathcal{N}(x, \sigma^2 I_{d_x})$. The factorization (1) is mathematically equivalent to setting up the noise process such that conditioned on the ligand x, the noisy ligand y is independent of the pocket ξ: $p(y|x, \xi) = p(y|x)$. Below we see that due to this conditional independence the empirical Bayes results used in (Saremi & Hyvärinen, 2019) readily generalize to our conditional setting.

In particular, we would like to relate $\hat{x}(y|\xi)$, the conditional least-squares Bayes estimator of X given (y, ξ), to the smoothed conditional score function $g(\cdot|\xi)$, defined by

$$g(y|\xi) = \nabla_y \log p(y|\xi). \tag{3}$$

Due to our noise process given by (1), the conditional Bayes estimator $\hat{x}(y|\xi) = \mathbb{E}[X|y, \xi]$ is given by

$$\hat{x}(y|\xi) = \frac{\int xp(y|x)p(x|\xi)dx}{\int p(y|x)p(x|\xi)dx}. \tag{4}$$

This leads to the following:

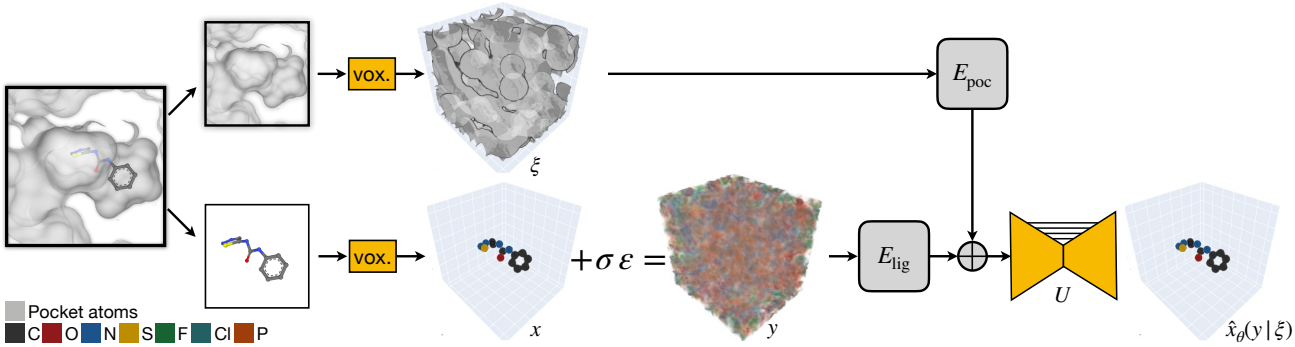

Figure 2. Conditional denoiser architecture. Given a ligand-pocket complex sample, we discretize each molecule resulting in the voxelized ligand x and pocket ξ. The ligand is corrupted by Gaussian noise with noise level σ. The corrupted ligand and pocket are encoded into a common embedding space (with the same spatial dimensions as the inputs) with encoders E_{lig} and E_{poc}, respectively. The two representations are added together and forwarded through a 3D U-Net U to recover the clean version of the ligand. To facilitate visualization, we threshold the grid values, $\hat{x} = \mathbb{1}_{\geq .1}(\hat{x})$.

Proposition 1. *Given the noise process* (1) *and* (2)*, the conditional Bayes estimator* (4) *can be written in closed form in terms of the conditional score function* (3)*:*

$$\hat{x}(y|\xi) = y + \sigma^2 g(y|\xi). \tag{5}$$

Proof. The derivation is simple and similar to the known classical results (Robbins, 1956), but we provide it here for completeness. We start with the following property of the Gaussian kernel (2):

$$\sigma^2 \nabla_y p(y|x) = (x - y)p(y|x).$$

We then multiply both sides of the above equation by $p(x|\xi)$ and integrate over x. Since ∇_y and the integration over x commute, using (1), we arrive at

$$\sigma^2 \nabla_y p(y|\xi) = \int x p(y|x) p(x|\xi) dx - y p(y|\xi).$$

Eq. (5) follows through by dividing both sides of the above equation by $p(y|\xi)$ and using (1), (3), and (4).[1] □

Due to this extension, summarized by Proposition 1, the (unconditional) neural empirical Bayes framework (Saremi & Hyvärinen, 2019) can be readily adopted for both learning the conditional score function $\nabla_y \log p(y|\xi)$ and drawing approximate samples from $p(x|\xi)$. By learning the conditional score function with a least-squares denoising objective we are able to sample from $p(y|\xi)$ using Langevin MCMC (*walk*), and also able to estimate clean ligands conditioned on the pocket using the Bayes estimator (*jump*), thus drawing approximate samples from $p(x|\xi)$. We refer to this sampling scheme as *conditional walk-jump sampling* (cWJS). The details on both learning and sampling are given next.

[1]The relation (5) between the conditional estimator and the conditional score function is the conditional form of what is referred to in the literature as Tweedie's formula, first derived for Gaussian kernels by Miyasawa (1961).

3.3. Conditional denoiser

From the perspective of learning $p(y|\xi)$, the key property of (5) is the appearance of the *score function* (Hyvärinen, 2005) in the right hand side. This means that in setting up the learning objective we do not have to worry about the intractable partition function (Saremi & Hyvärinen, 2019). In this paper we parametrize the conditional denoiser, *i.e.*, the left hand side of (5), from which the conditional score function is derived.

There are many ways to construct a conditional voxel denoiser. In this work we focus on a network design that is simple, scalable, and shows good empirical results. We follow the conditional image generation literature (see Sec. 2.3) and propose the following architecture: (i) encode the noisy ligand and the pocket with separate encoders, (ii) merge their representations and (iii) pass through an encoder-decoder architecture to predict clean sample. More precisely:

$$\hat{x}_\theta(y|\xi) = U_{\theta_3}(E_{\text{lig}}(y;\theta_1) + E_{\text{poc}}(\xi;\theta_2)), \tag{6}$$

where $\hat{x}_\theta : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\xi} \to \mathbb{R}^{d_x}$ is parameterized by $\theta = (\theta_1, \theta_2, \theta_3)$, E_{lig} and E_{poc} encode the noisy ligand y and pocket ξ to a common representation $\mathbb{R}^{d_e}$, $d_e = c_e L^3$, with the same spatial dimensions as input, and U is a 3D U-Net. Fig. 2 shows an overview of the model architecture.

Training samples are generated by first sampling a ligand-pocket complex from the empirical distribution and voxelize each of them, resulting in the pair (x, ξ). We add Gaussian noise drawn from $\mathcal{N}(0, \sigma^2 I_{d_x})$ to the ligand: $y = x + \sigma\,\varepsilon$, $\varepsilon \sim \mathcal{N}(0, I_{d_x})$. The denoising network (6) takes the pair (y, ξ) as input and should output x, the clean ligand. The model is trained by minimizing the mean square error over all voxels in all voxelized ligands, *i.e.*, by minimizing

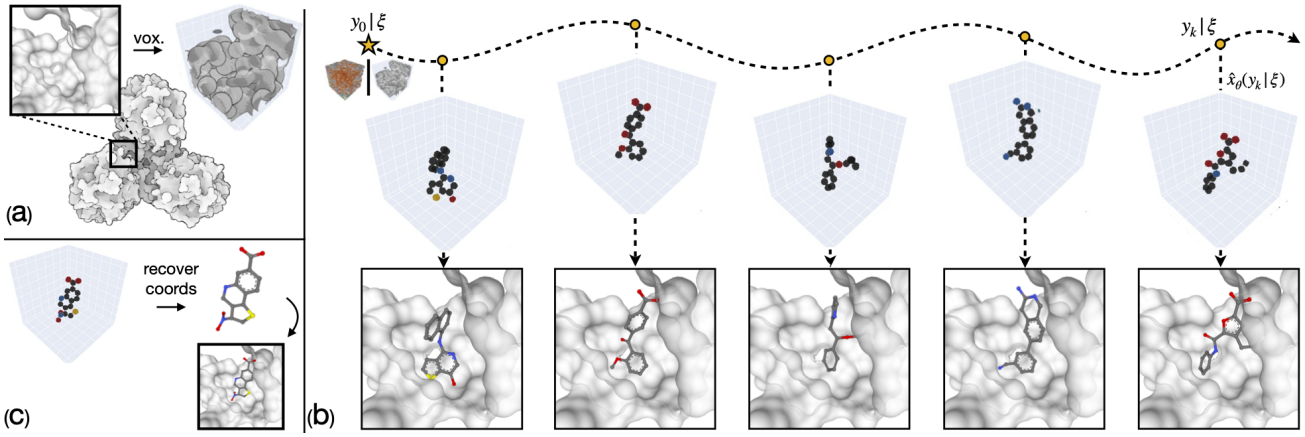

Figure 3. Illustration of pocket-conditional walk-jump sampling chain. (a) First, we voxelize a given protein binding pocket. (b) Then, we sample noisy voxelized ligands (given the pocket) with Langevin MCMC and estimate clean samples with the estimator. (c) Finally, we recover the atomic coordinates from voxel grids. In this figure, jumps are done at every $\Delta k = 100$ walk steps.

Algorithm 1 cWJS, using the BAOAB scheme (Sachs et al., 2017, Algorithm 1) adapted for the conditional ULD (9).

1: **Input** Learned conditional estimator $\hat{x}_\theta$, score function g_θ (from (8)), noise level σ, protein pocket ξ
2: **Input** step size δ, friction γ, steps taken K
3: **Output** $\widehat{x}$
4: $y_0 \sim \mathcal{N}(0, \sigma^2 I_{d_x}) + \mathcal{U}_{d_x}(0,1)$
5: $v_0 \leftarrow 0$
6: **for** $k = 0, \dots, K-1$ **do**
7: $\quad y_{k+1} \leftarrow y_k + \frac{\delta}{2} v_k$
8: $\quad g \leftarrow g_\theta(y_{k+1}|\xi)$
9: $\quad v_{k+1} \leftarrow v_k + \frac{\delta}{2} g$
10: $\quad \varepsilon \sim \mathcal{N}(0, I_{d_x})$
11: $\quad v_{k+1} \leftarrow \mathrm{e}^{-\gamma\delta} v_{k+1} + \frac{\delta}{2} g + \sqrt{(1-\mathrm{e}^{-2\gamma\delta})}\varepsilon$
12: $\quad y_{k+1} \leftarrow y_{k+1} + \frac{\delta}{2} v_{k+1}$
13: **end for**
14: $\hat{x} \leftarrow \hat{x}_\theta(y_K|\xi)$

the following loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{\substack{(x,\xi)\sim p(x,\xi),\\ \varepsilon\sim\mathcal{N}(0,I_{d_x})}} \left\| \hat{x}_\theta(x + \sigma\varepsilon|\xi) - x \right\|^2. \tag{7}$$

Finally, following learning, we can approximate the score function of the pocket-conditional distribution using (5):

$$g_\theta(y|\xi) = \frac{1}{\sigma^2}(\hat{x}_\theta(y|\xi) - y). \tag{8}$$

3.4. Conditional walk-jump sampling

We leverage the estimator (6), and the learned score function (8) to sample voxelized ligands conditioned on (voxelized) protein pockets. This is done by adapting the walk-jump scheme (Saremi & Hyvärinen, 2019; Pinheiro et al., 2023; Frey et al., 2024) to the conditional setting, resulting in the following two-step approach, which we call conditional walk-jump sampling (cWJS) (see Algorithm 1):

(i) *(walk step)* Sample noisy ligands from $p(y|\xi)$ with Langevin MCMC. We consider the conditional form of the underdamped Langevin diffusion (ULD):

$$\begin{aligned} dv_t &= -\gamma v_t dt + g_\theta(y_t|\xi)dt + \sqrt{2\gamma}\, dB_t, \\ dy_t &= v_t dt, \end{aligned} \tag{9}$$

where $(dB_t)_{t\geq 0}$ is the standard Brownian motion in $\mathbb{R}^{d_x}$ and γ is the friction. We use the discretization algorithm proposed by Sachs et al. (2017) to generate samples y_k (the inner loop of Algorithm 1).

(ii) *(jump step)* At an arbitrary time step K, typically after a burn-in time, we estimate clean molecules with the conditional estimator, *i.e.*, $\hat{x}_K = \hat{x}_\theta(y_K|\xi)$. Since jumps do not interact with the walks, this step can be moved to inner loop at the cost of memory (storing clean samples). More explanations are given below.

As we alluded to above, a key property of cWJS (which is borrowed from WJS) is that the jumps are decoupled from the walks (Saremi & Hyvärinen, 2019). Therefore, we can estimate samples at any arbitrary step of the MCMC chain. This is in contrast to diffusion models, where every sample requires a full MCMC chain to go from noise to sample. Fig. 3 illustrates the generation process starting from noise, estimating clean molecules (given the pocket) at each 100 steps.

Algorithm 1 describes a simple implementation of the ("*de novo*") conditional walk-jump sampling method: initialize a chain (lines 4-5), walk K steps (lines 6-13) and jump (line 14) to estimate sample $\hat{x}$, given pocket ξ. We use the discretization scheme by (Sachs et al., 2017, Algorithm 1),

Table 1. Results on CrossDocked2020 test set. Arrows ↑/↓ denote that higher/lower numbers are better, respectively. For the last column, numbers close to Reference are better. Baseline results are taken from (Guan et al., 2023b). For each metric, we **bold** and underline the best and second best methods, respectively. Our results are shown with mean/standard deviation across 3 runs. †This method uses different training assumptions and relies on (external) rule-based algorithms to generate samples.

	VinaScore (↓)		VinaMin (↓)		VinaDock (↓)		High aff. (%, ↑)		QED (↑)		SA (↑)		Diversity (↑)		# atoms / mol.	
	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.	Avg.	Med.
Reference	-6.36	-6.46	-6.71	-6.49	-7.45	-7.26	-	-	.48	.47	.73	.74	-	-	22.8	21.5
AR	-5.75	-5.64	-6.18	-5.88	-6.75	-6.62	37.9	31.0	.51	.50	.63	.63	.70	.70	17.6	16.0
Pocket2mol	-5.14	-4.70	-6.42	-5.82	-7.15	-6.79	48.4	51.0	.56	.57	**.74**	**.75**	.69	.71	17.7	15.0
DiffSBDD	-1.94	-4.24	-5.85	-5.94	-7.00	-6.90	46.3	47.2	.48	.48	.58	.57	.73	.72	24.0	23.0
TargetDiff	-5.47	-6.30	-6.64	-6.83	-7.80	-7.91	58.1	59.1	.48	.48	.58	.58	.72	.71	24.2	24.0
DecompDiff†	-5.67	-6.04	-7.04	-7.09	**-8.39**	**-8.43**	64.4	71.0	.45	.43	.61	.60	.68	.68	20.9	**21.0**
VoxBind$_{\sigma=0.9}$	**-6.94** (±.00)	**-7.11** (±.04)	**-7.54** (±.01)	**-7.55** (±.04)	-8.30 (±.02)	-8.41 (±0.1)	**71.3** (±2.4)	**71.5** (±.00)	**.57** (±.00)	**.58** (±.00)	.70 (±.00)	.69 (±.00)	.73 (±.00)	.74 (±.00)	**23.4** (±.1)	24.0 (±.0)
VoxBind$_{\sigma=1.0}$	-6.63 (±.03)	-6.70 (±.02)	-7.12 (±.02)	-7.18 (±.03)	-7.82 (±.01)	-7.89 (±0.01)	58.3 (±0.2)	61.5 (±.00)	.55 (±.03)	.57 (±.00)	.69 (±.00)	.69 (±.00)	**.75** (±.00)	**.76** (±.00)	21.7 (±.1)	**22.0** (±.0)

which requires the step size δ. Due to the decoupling of the jumps, line 14 in the algorithm can also be evaluate at the end of the inner loop $\hat{x}_{k+1} \leftarrow \hat{x}_\theta(y_{k+1}|\xi)$ (or preferably not at every step, but at some desired frequency Δk), *e.g.*, in Fig. 3 we set $\Delta k = 100$.

Finally, we run the same peak detection post-processing of VoxMol (Pinheiro et al., 2023) to recover the atomic coordinates from the generated voxelized ligand. See Sec. A.2 (appendix) for more implementation details on how we perform sampling.

4. Experiments

4.1. Experimental setup

Dataset. We benchmark our model on CrossDocked2020 (Francoeur et al., 2020), a popular dataset for pocket-conditional molecule generation. We follow previous work (Schneuing et al., 2022; Guan et al., 2023a;b) and use the pre-processing and splitting proposed by (Luo et al., 2021): the cross-docked dataset is reduced from 22.5M to 100,100 pairs of "high-quality" ligand-pocket pairs. The pockets are clustered (sequence identity $< 30\%$ using MMseqs2 (Steinegger & Söding, 2017)) and 100,000 and 100 ligand-pocket pairs are partitioned into the training and test sets, respectively. We take 100 samples from the training set as our hold-out validation set.

Ligands are represented with seven chemical elements (C, O, N, S, F, Cl and P) and protein pockets with four (C, O, N, S). Both molecules are modeled with implicit hydrogens. In the case of protein pockets, we consider all heavy atoms of every amino acid. Each ligand and its associated pocket are centered around the center of mass of the ligand and discretized on a cubic grid with length 64, resulting in tensors of dimensions $\mathbb{R}^{7\times64\times64\times64}$ and $\mathbb{R}^{4\times64\times64\times64}$ for ligand and pocket, respectively. During training, we apply random translation (uniform value between $[-1, 1]$ Å on 3D coordinates) and rotation (uniform value between $[0, 2\pi)$ on three Euler angles) to each ligand-protein training sample. These augmentations are applied to both molecules so that their relative pose remains unchanged.

Architecture. Encoders E_{lig} and E_{poc} map the voxelized molecules to a common embedding space with latent dimension $c_e = 16$ and same spatial dimensions, *i.e.*, in $\mathbb{R}^{d_e}, d_e = c_e \times 64^3$. The encoders contain two residual blocks, each having two padded $3 \times 3 \times 3$ convolutional layers with 16 units followed by SiLU non-linearities (Elfwing et al., 2018). The two embeddings are added together and the merged representation goes through a 3D U-Net architecture (Ronneberger et al., 2015) which predict the clean ligand, of dimension $d_x = 7 \times 64^3$. We use the same 3D U-Net architecture as in (Pinheiro et al., 2023), only modifying the dimensionality of the input and output. We train two versions of the model, one with noise level $\sigma = 0.9$ and another with $\sigma = 1.0$.

The models are trained with batch size of 64, learning rate 10^{-5} and weight decay 10^{-2}. The weights are updated with AdamW optimizer (Loshchilov & Hutter, 2019) ($\beta_1 = .9$, $\beta_2 = .999$) and exponential moving average with decay .999. Both models are trained for 340 epochs on four NVIDIA A100 GPUs (a total of ten days). For benchmarking purposes, we generate samples by repeatedly applying Algorithm 1, with $K = 500$ steps. We use $\delta = \sigma/2$ and fix $\gamma = 1/\delta$, *i.e.*, the "effective friction" (Saremi et al., 2023) is set to 1.

Baselines. We compare our method with various recent benchmarks for pocket-conditional ligand generation: two autoregressive models on point clouds (*AR* (Luo et al., 2021) and *Pocket2Mol* (Peng et al., 2022)) and three diffusion models applied on point clouds (*DiffSBDD* (Schneuing et al., 2022), *TargetDiff* (Guan et al., 2023a) and *DecompDiff* (Guan et al., 2023b)). Unlike other baselines, DecompDiff (the current state of the art) relies on a rule-based

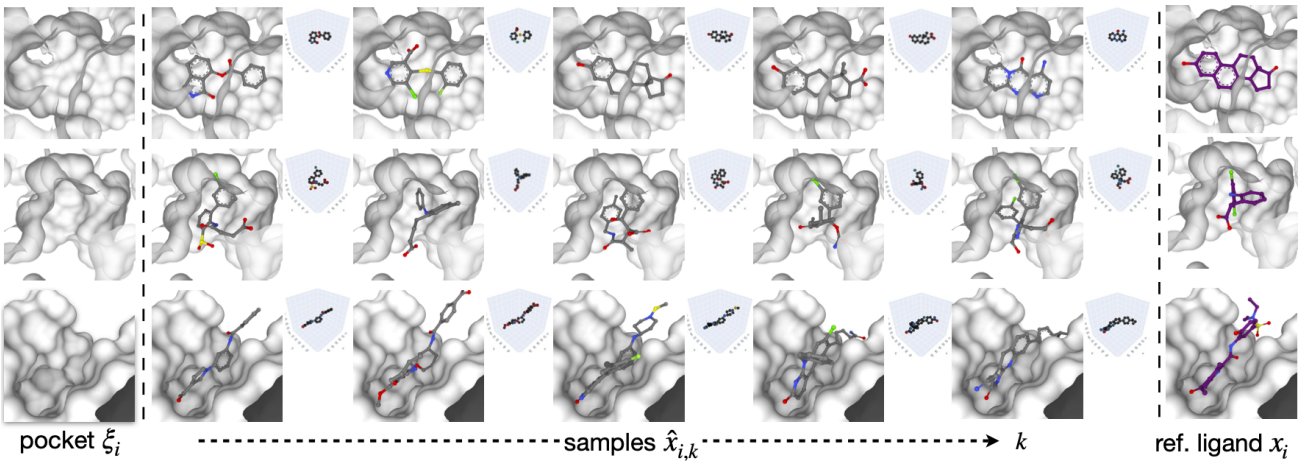

Figure 4. Example of generated ligands $\hat{x}_{i,k}$ given pocket ξ_i. Each row represents a single chain of samples for a given protein pocket (1E3R, 2I2Z, 5CRZ from top to bottom). For each generated sample, we show the ligand-pocket complex and the generated voxelized molecule. The samples from each row are generated from the same MCMC chain. The provided ground-truth ligands are shown on the last column.

algorithm (AlphaSpace2, proposed by Rooklin et al. (2015)) to compute subpockets and decomposable priors during sampling. All methods with the exception of DecompDiff—including VoxBind—uses OpenBabel (O'Boyle et al., 2011) to assign bonds from generated atom coordinates.

Metrics. We evaluate performance using similar metrics as previous work[2]. For each method, we sample 100 ligands for each of the 100 protein pockets. We measure affinity with three metrics using AutoDock Vina (Eberhardt et al., 2021): *VinaScore* computes the binding affinity (docking score) of the molecule on its original generated pose, *VinaMin* performs a local energy minimization on the ligand followed by dock scoring, *VinaDock* performs full re-docking (search and scoring) between ligand and target. *High affinity* computes the percentage of generated molecules that has lower VinaDock score than the reference ligand. We measure drug-likeness, *QED* (Bickerton et al., 2012), and synthesizability, *SA* (Ertl & Schuffenhauer, 2009), scores of generated molecules with RDKit (Landrum, 2016). *Diversity* is computed by averaging the Tanimoto distance (in RDKit fingerprints) of every pair of generated ligands per pocket (Bajusz et al., 2015). The *# atoms / molecules* is the average number of (heavy) atoms per molecules.

We also compute metrics proposed in PoseCheck (Harris et al., 2023)[3]. *Steric clash* computes the number of clashes between generated ligands and their associated pockets. *Strain energy* measures the difference between the internal energy of generated molecule's pose (without pocket) and a relaxed pose (computed using Universal Force Field (Rappé et al., 1992) within RDKit). *Interaction fingerprints* describe intramolecular interactions between the generated ligands and protein pockets. See (Harris et al., 2023) for more details about these metrics.

[2]We use the implementation of https://github.com/guanjq/targetdiff.

[3]Code available at https://github.com/cch1999/posecheck.

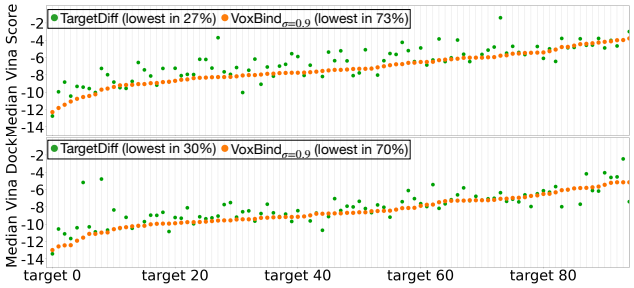

Figure 5. Median VinaScore and VinaDock (score on generated and redocked poses, respectively) of all generated molecules for each target on the test set (lower is better). Pockets are sorted by VoxBind$_{\sigma=0.9}$'s score.

4.2. Results

Binding affinity. Table 1 compares models on CrossDocked2020 *test set* in terms of binding affinity and molecular properties. For each metric, we compute the mean and the median over all generated molecules. The row *Reference* shows results of ground-truth ligands provided on the test set.

Our models achieve better results than TargetDiff (the best model trained under same assumptions) and DecompDiff (which relies on different training assumptions) in most metrics evaluated. The molecules generated by VoxBind$_{\sigma=0.9}$ and VoxBind$_{\sigma=1.0}$ have *better binding affinity*. In particular, our models generate molecules with *better pose* than

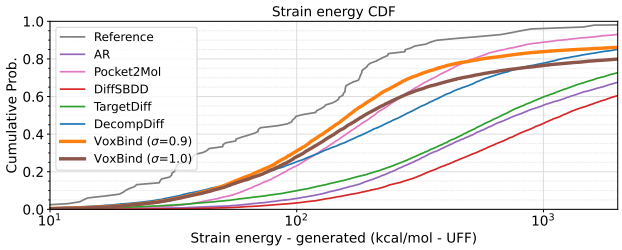

Figure 6. The cumulative distribution function of strain energy of molecules on their generated pose.

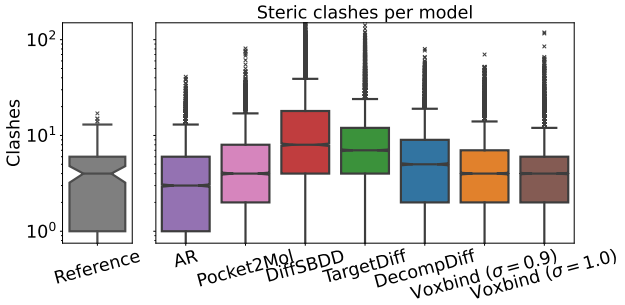

Figure 7. Number of steric clashes (lower is better) for the reference set and for molecules generated by each model.

other methods: their docking score with generated pose (VinaScore) is better/similar to that of methods after energy minimization (VinaMin).

Fig. 5 further compares our models with TargetDiff. It shows the median VinaScore and VinaDock (generated and redocked poses, respectively) of all generated molecules for each target on the test set. The generated molecules from VoxBind$_{\sigma=0.9}$ have better (computational) affinity on 73% of tested protein pockets.

Molecular properties. Table 1 also shows results on molecular properties. VoxBind achieves *better QED and SA* than diffusion-based models. The molecules generated by our models are particularly *more diverse* than baselines.

We note that our models *capture better the distribution of number of atoms per molecules* than the baselines. The autoregressive models generate molecules considerably smaller than the reference, while the diffusion model baselines tend to sample larger molecules. Our models achieve similar numbers as DecompDiff without the extra sampling assumptions. Unlike diffusion-based approaches, *our model captures this distribution implicitly and does not require sampling the number of atoms beforehand*.

We also compare models in terms of ring statistics on Fig. 4 (appendix): we show the histogram of number of rings per molecule (top), the histograms of ring sizes (*i.e.*, number of atoms per ring) (mid) and the histograms of fraction of aromatic atoms per molecule (bottom) for all generated molecules from different methods. In all cases, our models capture reference ring properties more favourably than baselines.

Molecular structures. First, we analyze the strain energies of molecules (as defined by PoseCheck (Harris et al., 2023)). Fig. 6 shows the cumulative distribution function (CDF) of strain energies (on the raw generate poses) for molecules from different models. The reference set has median strain energy of 102.5 kcal/mol, while the three best performing models, Pocket2Mol, VoxBind$_{\sigma=0.9}$ and VoxBind$_{\sigma=1.0}$ have 205.8, 161.9 and 188.3 kcal/mol, respectively. Fig. 11(a) (appendix) shows the CDF of strain energies after local minimization is performed on each molecule. The strain energies are largely reduced, but conclusions remain unchanged: VoxBind's molecules has *lower strain energies* than those generated by diffusion models. Additionally, Fig. 12 and Fig. 13 (appendix) show boxplots of strain energy per rotatable bonds, for the generated and minimized poses respectively, of molecules generated by different method.

Next, we measure the consistency of rigid fragments/substructures (*e.g.*, all carbons in a benzene ring should be in the same plane). We measure such consistency the same way as Guan et al. (2023a). First, we optmize each molecule with Merck molecular force field (MMFF) (Halgren, 1996). Then, we break the molecules into non-rotable fragments and, for each fragment, compute the RMSD between atoms coordinates before and after optimization. Fig. 11(b) (appendix) shows the RMSD for rigid fragments of different sizes. VoxBind models generate *rigid fragments that are more consistent* than those of other approaches.

Finally, we compare how models capture different atomic bond distances. Table 2 (appendix) shows the Jensen-Shannon divergence (JSD) between models and the reference set, for different types of bonds. We observe that VoxBind achieves lower JSD in most bond types, particularly double bonds and aromatic bonds. Moreover, we compare how models capture different atomic bond distances. Finally, Fig. 14 shows the frequency for the following four types of protein-ligand interactions considered by PoseCheck.

Steric clashes. Fig. 7 shows the number of steric clashes per ligand (on their generated poses). Our models generate ligands *with less clashes* than other methods (with the exception of AR, which perform poorly in all other metrics). In fact, VoxBind$_{\sigma=0.9}$ and VoxBind$_{\sigma=1.0}$ have mean clash score of 5.1 and 5.3, while AR, Pocket2Mol, DiffSBD, TargetDiff and DecompDiff have mean of 4.2, 5.8, 15.4, 10.8 and 7.1 clashes, respectively. The reference set, for comparison, has mean clash score of 4.7.

Sampling time. VoxBind is *more efficient* at sampling than other methods, sometimes an order of magnitude

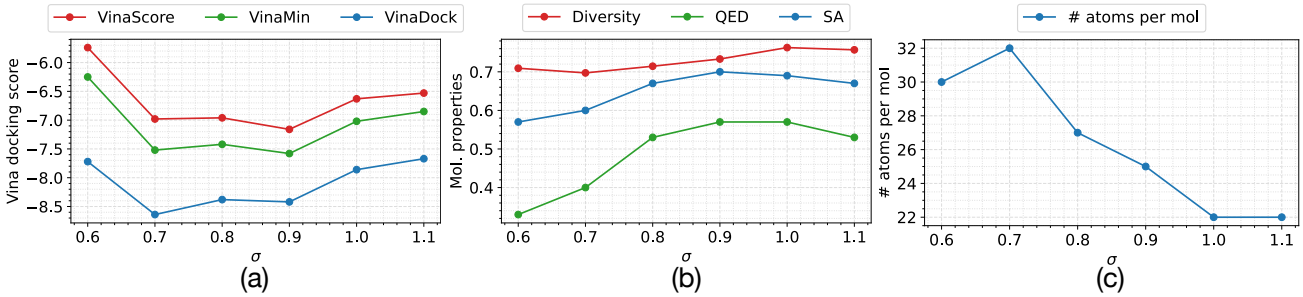

Figure 8. Effect of different noise level on (a) Vina metrics (VinaScore, VinaMin, VinaDock, lower is better), (b) molecular properties metrics (QED, SA and diversity, higher is better) and (c) number of atoms per molecules. In this experiment, we generate 100 molecules for the 100 pockets on validation set.

faster. VoxBind takes, on average, 492.2 seconds to generate 100 valid molecules with one A100 GPU, while Pocket2Mol, TargetDiff and DecompDiff requires 2,544, 3,428 and 6,189 seconds respectively (baselines running times taken from Guan et al. (2023b)).

Qualitative results. Fig. 4, and Fig. 15, Fig. 16 on appendix show examples of generated ligands from our model.

4.3. Ablation studies

Noise level. The noise level is an important hyperparameter of our model. As the noise level increases, it becomes easier to sample from the smoothed distribution. However, denoising becomes harder. To find the best empirical noise level, we train our model with different noise levels (all other hyperparameters are kept the same). Then, we compare the quality of the samples conditioned on pockets from the *validation* set. Fig. 8 shows how different metrics change as the noise level changes. We found that $\sigma = .9$ and $\sigma = 1.0$ achieve the best results on the validation set and chose to report results with those two levels.

Data augmentation. We also train a version of VoxBind without data augmentation. In terms of molecular properties (QED and SA), the performance is very similar. However, we see a big difference in terms of binding affinity metrics. Fig. 9 (appendix) shows the median VinaScore of VoxBind trained with and without data augmentation for each target (we use $\sigma = 0.9$ in this experiment). The model trained with data augmentation has better affinity score on the generated poses on 85% of the targets.

5. Conclusion

we present VoxBind, a new score-based generative model for SBDD. We extended the neural empirical Bayes formalism and the walk-jump sampling algorithm to the conditional setting (cWJS) and show that our model outperforms previous work on an extensive number of computational metrics on a popular benchmark, while being faster to generate samples.

Our approach is flexible and allows us to adapt sampling to many practical SBDD settings, without any additional training. For example, if we start with a ligand that binds to the pocket of interest, we can initialize the MCMC chain with a smooth version of that ligand instead of noise.[4]

The flexibility and expressivity of 3D U-Nets comes at the cost of increased memory consumption. Therefore, the volume in 3D space that we can process is bounded by GPU memory. As shown empirically, our approach works well for drug-like molecule generation. However, more work (*e.g.*, on data representation and architecture) needs to be done to scale generation to larger molecules like nucleic acids and proteins. Additional future work includes better modeling of synthetic accessibility or integrating pocket dynamics into the generation process.

Impact statement

Structure-based drug design is an important component in modern drug discovery research and development. This is a very long and challenging endeavor that involves many steps. In this paper we propose a new pocket-conditional generative model, which deals with one of these steps. There is still a lot of work that need to be done to validate these kinds of models in practice (*e.g.*, wet-lab experimental validation, clinical trials, etc). That been said, if successful, advances in this field can directly impact quality of human health. Like many other modern powerful technologies, we need to ensure that these models are deployed in ways that are safe, ethical, accountable and exclusively beneficial to society.

Acknowledgements

The authors are grateful to Charlie Harris for assistance in running the Posecheck benchmark. We also thank the Prescient Design team for helpful discussions and Genentech's HPC team for providing a reliable environment to train/analyse models.

[4]We can easily adapt the sampling to other applications, *e.g.*, scaffold hopping or linking, by initializing the chains with molecular sub-parts fragments. These are related to "in-painting" tasks in computer vision.

References

Adams, K. and Coley, C. W. Equivariant shape-conditioned generation of 3D molecules for ligand-based drug design. *arXiv:2210.04893*, 2022.

Amit, T., Nachmani, E., Shaharbany, T., and Wolf, L. Segdiff: Image segmentation with diffusion probabilistic models. *arXiv:2112.00390*, 2021.

Anderson, A. C. The process of structure-based drug design. *Chemistry & biology*, 2003.

Bai, Y., Mei, J., Yuille, A. L., and Xie, C. Are transformers more robust than cnns? *NeurIPS*, 2021.

Bajusz, D., Rácz, A., and Héberger, K. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 2015.

Batatia, I., Kovacs, D. P., Simm, G., Ortner, C., and Csányi, G. Mace: Higher order equivariant message passing neural networks for fast and accurate force fields. *Neurips*, 2022.

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., and Hopkins, A. L. Quantifying the chemical beauty of drugs. *Nature chemistry*, 2012.

Blundell, T. L. Structure-based drug design. *Nature*, 1996.

Eberhardt, J., Santos-Martins, D., Tillack, A. F., and Forli, S. Autodock vina 1.2. 0: New docking methods, expanded force field, and python bindings. *JCIM*, 2021.

Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 2018.

Ertl, P. and Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 2009.

Fink, T., Bruggesser, H., and Reymond, J.-L. Virtual exploration of the small-molecule chemical universe below 160 daltons. *Angewandte Chemie International Edition*, 2005.

Flam-Shepherd, D. and Aspuru-Guzik, A. Language models can generate molecules, materials, and protein binding sites directly in three dimensions as xyz, cif, and pdb files. *arXiv:2305.05708*, 2023.

Francoeur, P. G., Masuda, T., Sunseri, J., Jia, A., Iovanisci, R. B., Snyder, I., and Koes, D. R. Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design. *Journal of chemical information and modeling*, 2020.

Frey, N. C., Berenberg, D., Kleinhenz, J., Hotzel, I., Lafrance-Vanasse, J., Kelly, R. L., Wu, Y., Rajpal, A., Ra, S., Bonneau, R., Cho, K., Loukas, A., Gligorijevic, V., and Saremi, S. Protein discovery with discrete walk-jump sampling. In *ICLR*, 2024.

Gebauer, N., Gastegger, M., and Schütt, K. T. Generating equilibrium molecules with deep neural networks. *arXiv:1810.11347*, 2018.

Gebauer, N., Gastegger, M., and Schütt, K. Symmetry-adapted generation of 3D point sets for the targeted discovery of molecules. In *NeurIPS*, 2019.

Geiger, M. and Smidt, T. E3nn: Euclidean neural networks. *arXiv:2207.09453*, 2022.

Gruver, N., Finzi, M., Goldblum, M., and Wilson, A. G. The Lie derivative for measuring learned equivariance. *arXiv:2210.02984*, 2022.

Guan, J., Qian, W. W., Peng, X., Su, Y., Peng, J., and Ma, J. 3D equivariant diffusion for target-aware molecule generation and affinity prediction. *ICLR*, 2023a.

Guan, J., Zhou, X., Yang, Y., Bao, Y., Peng, J., Ma, J., Liu, Q., Wang, L., and Gu, Q. DecompDiff: Diffusion models with decomposed priors for structure-based drug design. In *ICML*, 2023b.

Halgren, T. A. Merck molecular force field. i. basis, form, scope, parameterization, and performance of mmff94. *Journal of computational chemistry*, 1996.

Harris, C., Didi, K., Jamasb, A. R., Joshi, C. K., Mathis, S. V., Lio, P., and Blundell, T. Benchmarking generated poses: How rational is structure-based drug design with generative models? *arXiv:2308.0741*, 2023.

Hoogeboom, E., Satorras, V. G., Vignac, C., and Welling, M. Equivariant diffusion for molecule generation in 3D. In *ICML*, 2022.

Huang, L., Zhang, H., Xu, T., and Wong, K.-C. Mdm: Molecular diffusion model for 3D molecule generation. *arXiv:2209.05710*, 2022.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *JMLR*, 2005.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *ICLR*, 2014.

Köhler, J., Klein, L., and Noé, F. Equivariant flows: exact likelihood generative learning for symmetric densities. In *ICML*, 2020.

Landrum, G. Rdkit: Open-source cheminformatics software, 2016. URL https://github.com/rdkit/rdkit/releases/tag/Release_2016_09_4.

Li, L., Li, C., and Alexov, E. On the modeling of polar component of solvation energy using smooth gaussian-based dielectric function. *Journal of Theoretical and Computational Chemistry*, 2014.

Liu, M., Luo, Y., Uchino, K., Maruhashi, K., and Ji, S. Generating 3D molecules for target protein binding. *arXiv*, 2022.

Long, S., Zhou, Y., Dai, X., and Zhou, H. Zero-shot 3d drug design by sketching and generating. *NeurIPS*, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *ICLR*, 2019.

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Van Gool, L. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022.

Luo, S., Guan, J., Ma, J., and Peng, J. A 3D generative model for structure-based drug design. *NeurIPS*, 2021.

Luo, Y. and Ji, S. An autoregressive flow model for 3D molecular geometry generation from scratch. In *ICLR*, 2022.

Lyu, J., Wang, S., Balius, T. E., Singh, I., Levit, A., Moroz, Y. S., O'Meara, M. J., Che, T., Algaa, E., Tolmachova, K., et al. Ultra-large library docking for discovering new chemotypes. *Nature*, 2019.

Miyasawa, K. An empirical Bayes estimator of the mean of a normal population. *Bull. Inst. Internat. Statistics*, 1961.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, 2019.

O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., and Hutchison, G. R. Open babel: An open chemical toolbox. *Journal of cheminformatics*, 2011.

Orlando, G., Raimondi, D., Duran-Romaña, R., Moreau, Y., Schymkowitz, J., and Rousseau, F. Pyuul provides an interface between biological structures and deep learning algorithms. *Nature communications*, 2022.

Peng, X., Luo, S., Guan, J., Xie, Q., Peng, J., and Ma, J. Pocket2mol: Efficient molecular sampling based on 3D protein pockets. In *ICML*, 2022.

Pinheiro, P. O., Rackers, J., Kleinhenz, J., Maser, M., Mahmood, O., Watkins, A. M., Ra, S., Sresht, V., and Saremi, S. 3D molecule generation by denoising voxel grids. In *NeurIPS*, 2023.

Powers, A. S., Yu, H. H., Suriana, P., Koodli, R. V., Lu, T., Paggi, J. M., and Dror, R. O. Geometric deep learning for structure-based ligand design. *ACS Central Science*, 2023.

Pozdnyakov, S. N. and Ceriotti, M. Incompleteness of graph convolutional neural networks for points clouds in three dimensions. *arXiv:2201.07136*, 2022.

Ragoza, M., Masuda, T., and Koes, D. R. Learning a continuous representation of 3D molecular structures with deep generative models. In *Neurips, Structural Biology workshop*, 2020.

Ragoza, M., Masuda, T., and Koes, D. R. Generating 3D molecules conditional on receptor binding sites with deep generative models. *Chemical science*, 2022.

Rappé, A. K., Casewit, C. J., Colwell, K., Goddard III, W. A., and Skiff, W. M. Uff, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American chemical society*, 1992.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *ICML*, 2015.

Robbins, H. E. An empirical Bayes approach to statistics. In *Proc. 3rd Berkeley Symp. Math. Statist. Probab., 1956*, 1956.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

Rooklin, D., Wang, C., Katigbak, J., Arora, P. S., and Zhang, Y. Alphaspace: Fragment-centric topographical mapping to target protein–protein interaction interfaces. *Journal of chemical information and modeling*, 2015.

Rozenberg, E., Rivlin, E., and Freedman, D. Structure-based drug design via semi-equivariant conditional normalizing flows. In *ICLR, Machine Learning for Drug Discovery workshop*, 2023.

Sachs, M., Leimkuhler, B., and Danos, V. Langevin dynamics with variable coefficients and nonconservative forces: from stationary states to numerical methods. *Entropy*, 2017.

Saharia, C., Chan, W., Chang, H., Lee, C., Ho, J., Salimans, T., Fleet, D., and Norouzi, M. Palette: Image-to-image diffusion models. In *SIGGRAPH*, 2022a.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022b.

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. Image super-resolution via iterative refinement. *PAMI*, 2022c.

Saremi, S. and Hyvärinen, A. Neural empirical Bayes. *JMLR*, 2019.

Saremi, S., Srivastava, R. K., and Bach, F. Universal smoothed score functions for generative modeling. *arXiv:2303.11669*, 2023.

Satorras, V. G., Hoogeboom, E., and Welling, M. E (n) equivariant graph neural networks. In *ICML*, 2021.

Saxena, S., Herrmann, C., Hur, J., Kar, A., Norouzi, M., Sun, D., and Fleet, D. J. The surprising effectiveness of diffusion models for optical flow and monocular depth estimation. *arXiv:2306.01923*, 2023.

Schneuing, A., Du, Y., Harris, C., Jamasb, A., Igashov, I., Du, W., Blundell, T., Lió, P., Gomes, C., Welling, M., et al. Structure-based drug design with equivariant diffusion models. *arXiv:2210.13695*, 2022.

Skalic, M., Jiménez, J., Sabbadin, D., and De Fabritiis, G. Shape-based generative modeling for de novo drug design. *Journal of chemical information and modeling*, 2019.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Steinegger, M. and Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 2017.

Thomas, M., Bender, A., and de Graaf, C. Integrating structure-based approaches in generative molecular design. *Current Opinion in Structural Biology*, 2023.

Townshend, R. J., Vögele, M., Suriana, P., Derry, A., Powers, A., Laloudakis, Y., Balachandar, S., Jing, B., Anderson, B., Eismann, S., et al. Atom3D: Tasks on molecules in three dimensions. *NeurIPS*, 2020.

Vignac, C., Osman, N., Toni, L., and Frossard, P. Midi: Mixed graph and 3D denoising diffusion for molecule generation. *ICLR, MLDD workshop*, 2023.

Wang, L., Bai, R., Shi, X., Zhang, W., Cui, Y., Wang, X., Wang, C., Chang, H., Zhang, Y., Zhou, J., et al. A pocket-based 3D molecule generative model fueled by experimental electron density. *Scientific reports*, 2022a.

Wang, M., Hsieh, C.-Y., Wang, J., Wang, D., Weng, G., Shen, C., Yao, X., Bing, Z., Li, H., Cao, D., et al. Relation: A deep generative model for structure-based de novo drug design. *Journal of Medicinal Chemistry*, 2022b.

Wang, Y., Elhag, A. A., Jaitly, N., Susskind, J. M., and Bautista, M. A. Generating molecular conformer fields. *arXiv:2311.17932*, 2023.

Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. S. 3D steerable cnns: Learning rotationally equivariant features in volumetric data. *NeurIPS*, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *ICLR*, 2019.

Xu, M., Powers, A., Dror, R., Ermon, S., and Leskovec, J. Geometric latent diffusion models for 3D molecule generation. In *ICML*, 2023.

Zhang, Z., Min, Y., Zheng, S., and Liu, Q. Molecule generation for target protein binding with structural motifs. In *ICLR*, 2023.

A. Additional implementation details

A.1. Voxelized molecules

We represent molecules as voxelized atomic densities. We follow the same approach as (Pinheiro et al., 2023), as it has been shown to work well in practice for drug-like molecule generation. We apply the same approach for ligands and protein pockets. First, we convert each atom (of each molecule) into 3D Gaussian-like densities:

$$V_a(d, r_a) = \exp\Big(- \frac{d^2}{(.93 \cdot r_a)^2} \Big), \tag{10}$$

where V_a is defined as the fraction of occupied volume by atom a of radius r_a at distance d from its center. We use radius $r_a = .5$ for all atom on ligand molecules (as in (Pinheiro et al., 2023)) and use the Van der Waals radii for pocket atoms. Then, we compute the occupancy of each voxel in the grid:

$$\text{Occ}_{i,j,k} = 1 - \prod_{n=1}^{N_a} \big(1 - V_{a_n}(\|C_{i,j,k} - x_n\|, r_{a_n})\big), \tag{11}$$

where $\{a_n\}_{n=1}^{N_a}$ are the atoms of the molecule, $C_{i,j,k}$ are the coordinates (i,j,k) in the grid and x_n is the coordinates of the center of atom n (Li et al., 2014). Each atom type occupy a different channel grid (similar to R,G,B channels of images) and they take values between 0 and 1. We use use a grid with 64^3 voxels with resolution of .25Å per voxel. We model hydrogens implicitly and consider seven chemical elements for ligands (C, O, N, S, F, Cl and P) and four for pockets (C, O, N, S). This results in voxel grids with dimensions $d_x = 7 \times 64 \times 64 \times 64$ and $d_z = 4 \times 64 \times 64 \times 64$ for ligand and protein protein pockets, respectively.

Every ligand and its pocket pair are centered around the center of the mass of the ligand. During training, each training sample is augmented random augmentation (applied to both ligand and pocket). These augmentations are made of random translation (uniform value between [-1,1] on 3D coordinates) and rotation (uniform value between [0, 2π) on three Euler angles). These augmentations are applied on the point cloud before voxelizing them. We use the python package PyUUL (Orlando et al., 2022) to generate the voxel grids from the raw molecules (`.sdf` or `.pdb` format).

A.2. Sampling

The walk-jump sampling approach is very flexible and allows us to configure sampling in different ways. For example, we can chose the number of walk steps between jumps, the maximum number of walk steps per chain or the number of chains run in parallel. Different sampling hyperparameters can change the statistics of samples, *e.g.*, we increase sampling speed and reduce diversity if we reduce the number of walk steps between jumps.

Therefore, we decided to fix a set of sampling hyperparameters for benchmark purposes. In all our experiments we generate samples in the following way (100 chains in parallel): (i) initialize a chain y_0 (from noise and a pocket) and walk 400 Langevin MCMC *warm-up* steps to get to y_{400}, (ii) create a batch (size 100) with copies of the tensor y_{400} (iii) *walk* 100 steps then *jump* to estimate a clean molecule at step $\hat{x}_{500}$, (iv) return to step (i) and repeat until generate 100 valid samples. When sampling conditioned on pocket and ligand, we take 50 warm-up steps and 50 walk steps. We found this setting to provide a good trade-off in terms of performance/speed on validation set. However, it is by no means optimal and results can possibly be further improved by finding a better sampling recipe.

We extract the atomic coordinates from the generated voxelized grids following the same approach as (Pinheiro et al., 2023): we set all voxel values less than 0.1 to 0 then run a peak finding algorithm to get the 3D coordinates of each atom. The identity of the atom is the channel in the voxel grid. Once we have the set of atoms (types and coordinates), we follow previous work and use OpenBabel (O'Boyle et al., 2011) to assign bonds to the atoms.

The qualitative samples in this paper are sampled with 400 warm-up step followed by 100 steps for 10,000 steps on a *single* MCMC chain.

B. Additional results

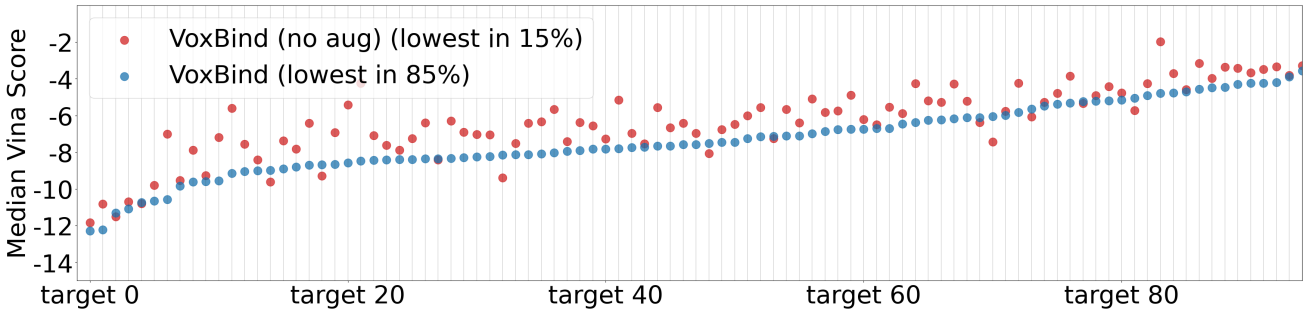

Figure 9. Median VinaScore of all generated molecules for each target on the test set (sorted by VoxBind's score). We compare VoxBind trained with and without data augmentation with $\sigma = 0.9$.

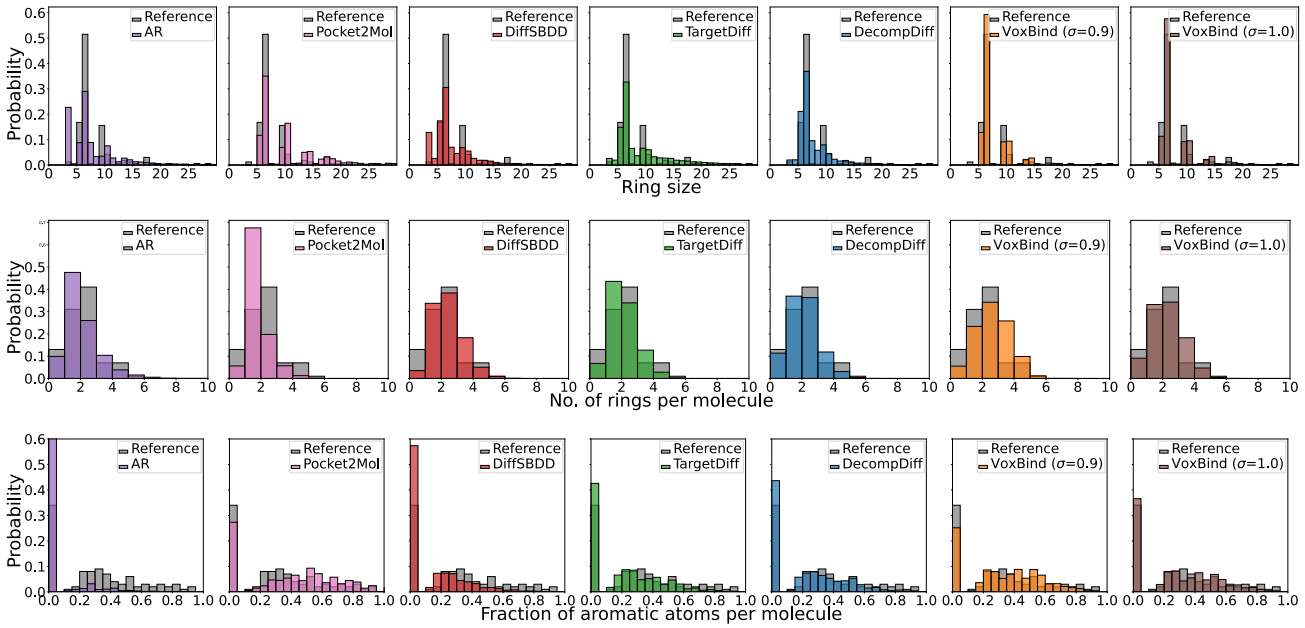

Figure 10. Histograms showing (top) distributions ring sizes for all rings, (mid) distributions of number of rings per molecule and (bottom) distributions of fraction of aromatic atoms per molecule in all generated molecules from different methods. The underlying distribution from the reference is also shown in each plot.

Table 2. Jensen-Shannon divergence (↓) between reference and generated molecules' distributions of bond distances. The symbols "– ", "=", ":" represent single bond, double bond and aromatic bond, respectively. For each metric, we **bold** and underline the best and second best methods, respectively.

	C — C	C = C	C — N	C = N	C — O	C = O	C : C	C : N
AR	.609	.620	.474	.635	.492	.558	.451	.552
Pocket2Mol	.496	.561	.416	.629	.454	.516	.416	.487
TargetDiff	.369	**.505**	.363	.550	.421	.461	.263	.235
DecompDiff†	<u>.359</u>	.537	**.344**	.584	<u>.376</u>	.374	.251	.269
VoxBind$_{\sigma=0.9}$	.372	<u>.528</u>	<u>.351</u>	<u>.528</u>	.400	**.326**	<u>.215</u>	**.186**
VoxBind$_{\sigma=1.0}$	**.357**	.533	.354	**.418**	**.354**	<u>.335</u>	**.210**	<u>.191</u>

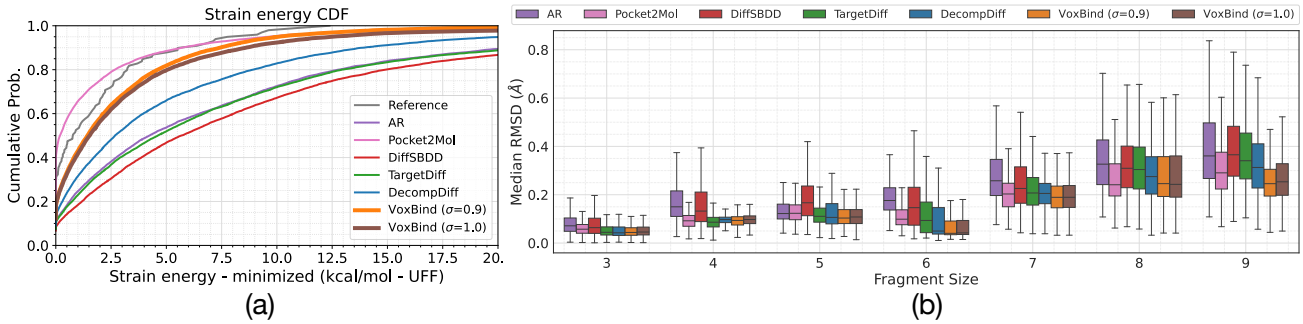

Figure 11. (a) The cumulative distribution function of strain energy of molecules after small force-field relaxation. As expected, the strain energies in this setting are much lower than with the raw generated poses. (b) Median RMSD (↓) between fragments of generated molecules before and after force-field optimization.

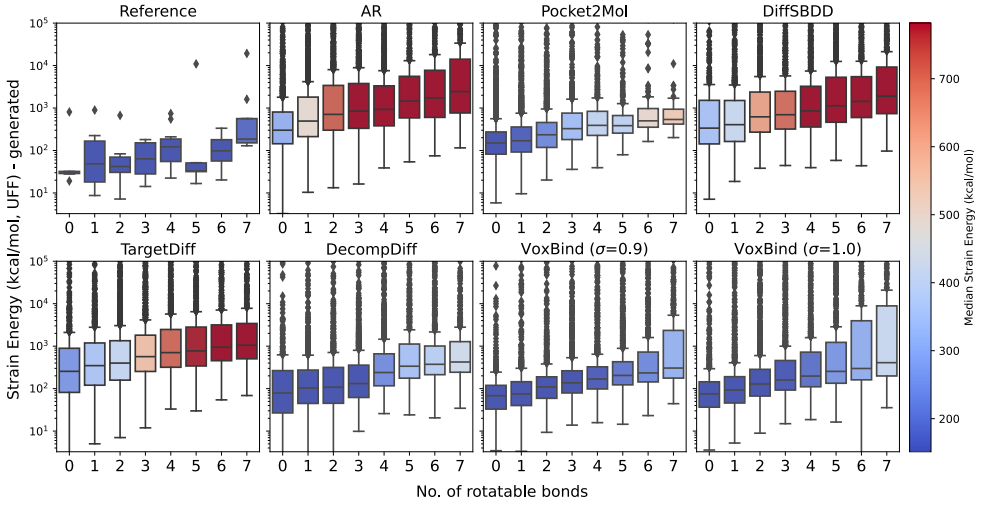

Figure 12. Boxplots of strain energies (lower is better) of generated molecules (on their generated poses) per number of rotatable bonds for all methods. Box color shows median strain value.

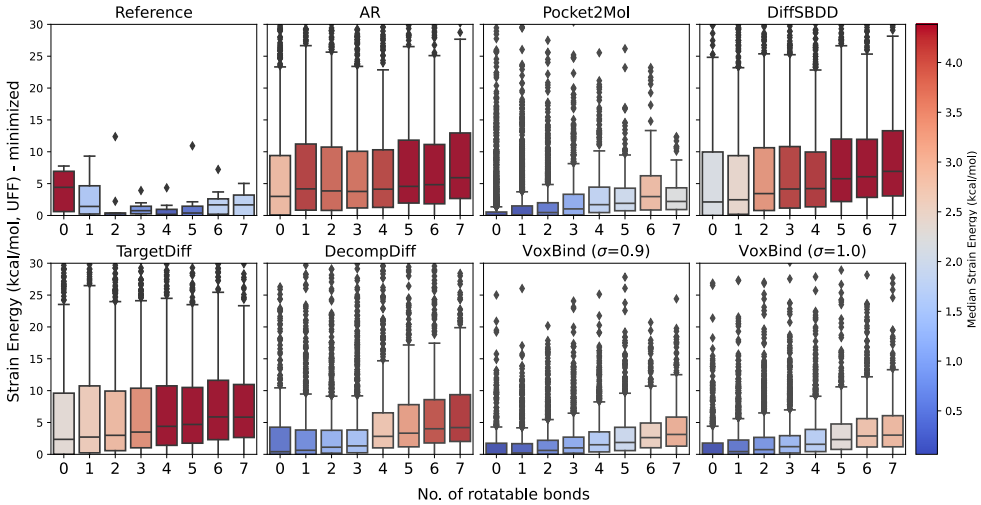

Figure 13. Boxplots of strain energies (lower is better) of generated molecules (after local minimization) per number of rotatable bonds for all methods. Box color shows median strain value.

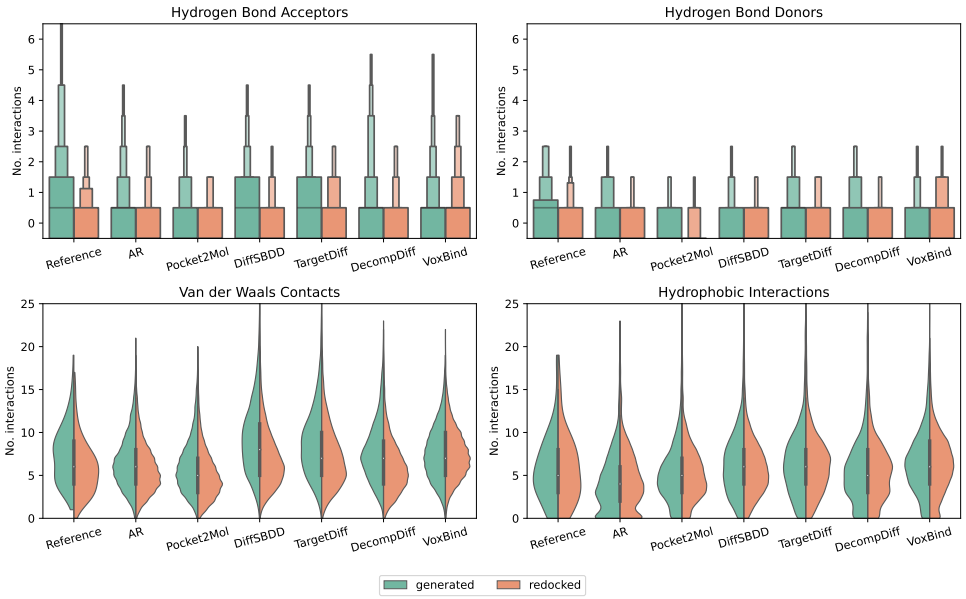

Figure 14. Protein-ligand interactions in generated poses (green) and redocked poses (orange). The frequency of (a) hydrogen bond acceptors, (b) hydrogen bond donors, (c) Van der Waals contacts and (d) hydrophobic interactions are shown.

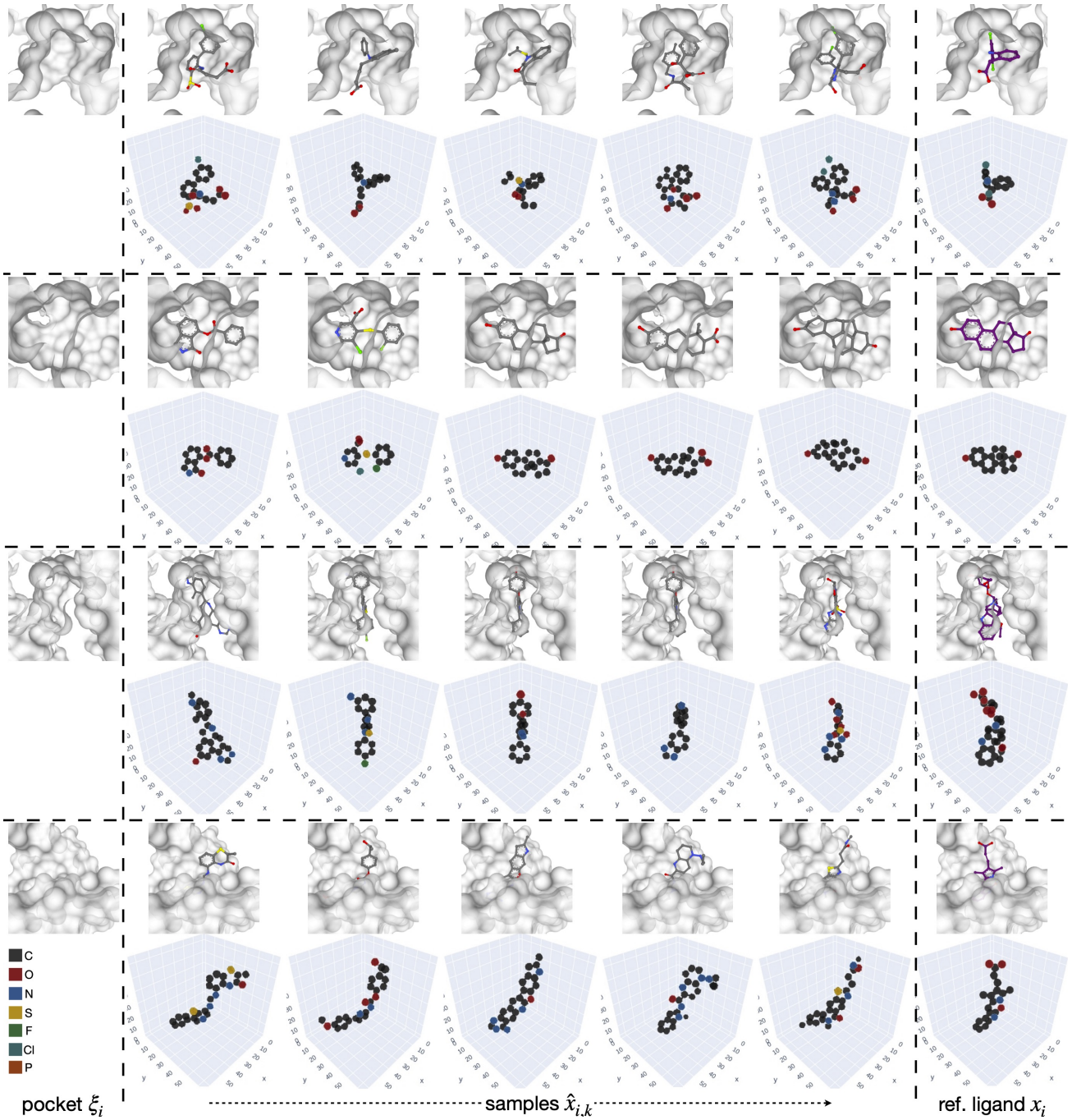

Figure 15. Example of generated ligands $\hat{x}_{i,k}$ given pocket ξ_i. Each row represents a single chain of samples for a given protein pocket (`2i2z`, `1ogx`, `3u57`, `4jlc` from top to bottom). The samples from each row are generated from the same MCMC chain. The provided ground-truth ligands are shown on the last column.

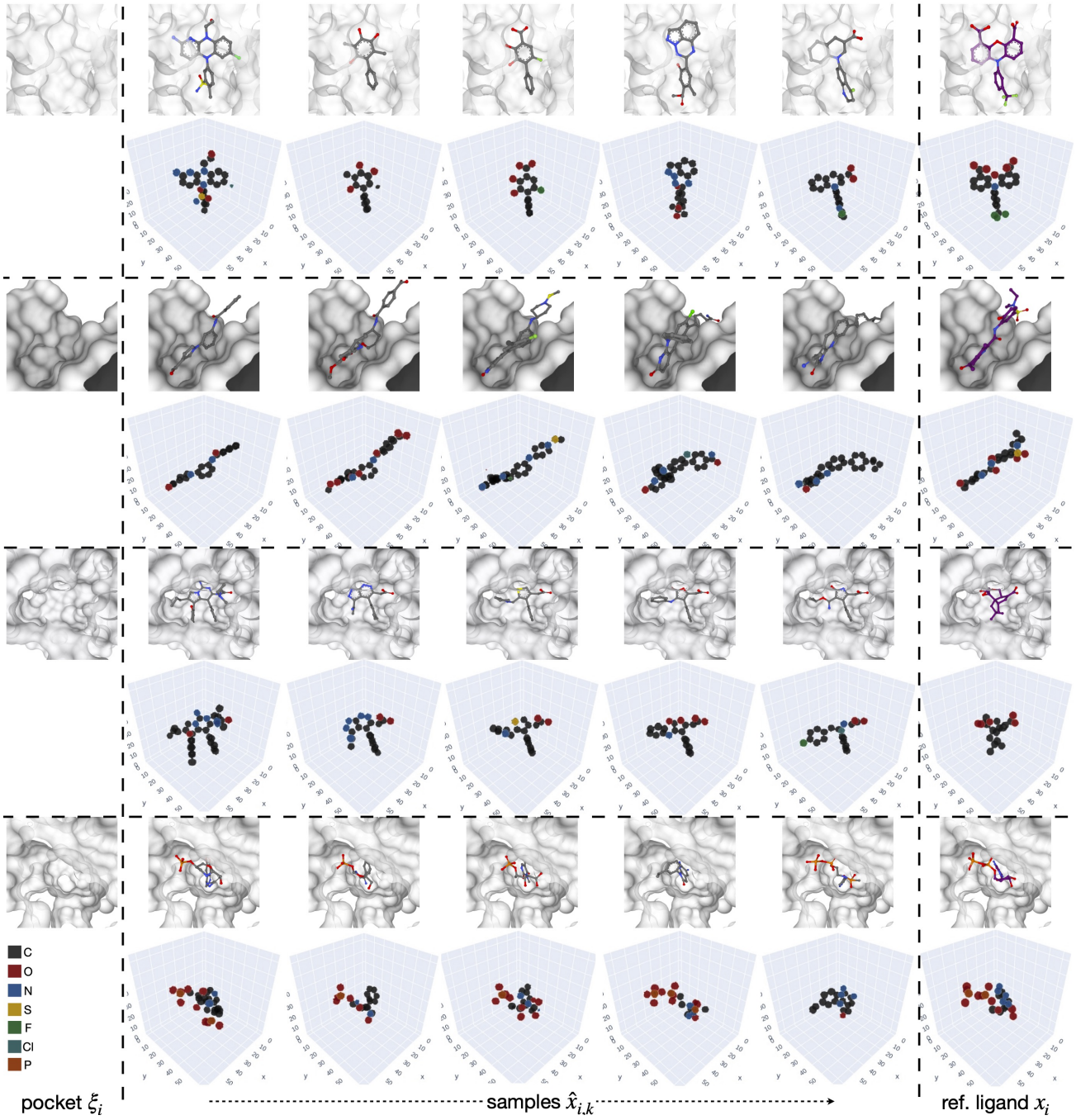

Figure 16. Example of generated ligands $\hat{x}_{i,k}$ given pocket ξ_i. Each row represents a single chain of samples for a given protein pocket (`5aks`, `5crz`, `5l1v`, `1e8h` from top to bottom). The samples from each row are generated from the same MCMC chain. The provided ground-truth ligands are shown on the last column.