Autoregressive fragment-based diffusion for pocket-aware ligand design

Mahdi Ghorbani
University of California, San Francisco
ghorbani@keiserlab.org

Leo Gendelev
Genentech
gendelev.leo@gene.com

Paul Beroza
Genentech
berozap@gene.com

Michael J. Keiser
University of California, San Francisco
keiser@keiserlab.org

Abstract

In this work, we introduce AutoFragDiff, a fragment-based autoregressive diffusion model for generating 3D molecular structures conditioned on target protein structures. We employ geometric vector perceptrons to predict atom types and spatial coordinates of new molecular fragments conditioned on molecular scaffolds and protein pockets. Our approach improves the local geometry of the resulting 3D molecules while maintaining high predicted binding affinity to protein targets. The model can also perform scaffold extension from user-provided starting molecular scaffold.

1 Introduction

Rational drug design against defined binding pockets relies heavily on computational modeling.Stokes et al. [2020], Anderson [2003] Traditionally, the diversity of small-molecule candidates and the high degrees of freedom inherent in ligand-protein binding systems make navigating chemical space computationally intensive.Lipinski and Hopkins [2004] Moreover, target-aware molecular design strives to balance optimizing for potency against specific target structures while maintaining desirable absorption, distribution, metabolism, and excretion (ADME) and pharmacokinetic and pharmacodynamic (PKPD) properties.Skalic et al. [2019] While many target protein structures are available, effectively harnessing this information to design novel drug-like compounds with desired therapeutic effects remains an active area of research.Ragoza et al. [2017]

Diffusion models Ho et al. [2020], Kingma et al. [2021] generate 3D molecular structures from underlying distributions of molecular dataHoogeboom et al. [2022], thus enabling the generation of diverse molecular candidates that reflect real chemical space. However, these models struggle to capture the nuances of local molecular geometry. Specifically, maintaining the correct spatial arrangements and conformations of functional groups and atoms remains challenging.Harris et al. [2023] While the overall structure might resemble known molecules, minor deviations in local geometry can significantly impact the bioactivity and specificity of the generated compounds.

Many pocket-specific molecule generation models have leveraged autoregressive strategies. In these models, atoms are placed individually, and bonds are determined separately. Drotár et al. [2021], Liu et al. [2022]. However, this sequential approach can be cumbersome and error-prone; even generating a benzene ring is a laborious six-step procedure. Fragment-based generation strategies sidestep some of these drawbacks. Our work employs Autoregressive Diffusion Models (ARDMs),Hoogeboom et al. [2021] which can generate data in a flexible order. This unique feature enables ARDMs to bridge the gap between order-agnostic autoregressive and diffusion-based generative models.Igashov et al.

NeurIPS 2023 Generative AI and Biology Workshop.

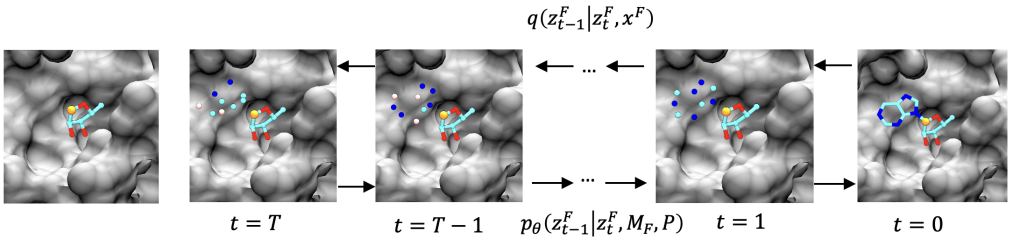


Figure 1: Noising and Sampling for a single fragment inside a protein pocket. Yellow spheres show the anchor point.

[2022], Hoogeboom et al. [2022], Schneuing et al. [2022], Guan et al. [2023] Unlike their traditional counterparts, ARDMs don't adhere to strict architectural norms for neural networks, yet they achieve comparable results in fewer steps.

In this study, we combine fragment-based drug design with autoregressive diffusion models. Unlike traditional autoregressive methods that work atom by atom, this combined approach allows each fragment to undergo a denoising process, predicting atom coordinates and atom types (Figure 1). Rather than relying on a fixed fragment library, our approach dynamically generates fragments, providing flexibility in the diversity of fragments produced. This approach generates molecules with more accurate local geometries for pocket-based molecule generation, delivering greater precision and efficiency in drug design.

2 Related Work

Generative models and geometric deep learning have influenced recent pocket-based drug design. Atz et al. [2021], Bronstein et al. [2017]. Li et al. [2021] introduced an autoregressive generative model designed to sample ligands, using the pocket as a conditioning constraint. Building on this work, Peng et al. Peng et al. [2022b] introduced Pocket2Mol, which uses an E(3) equivariant graph neural networkSatorras et al. [2021] that accounts for rotation and translation symmetries in 3D space for more accurate molecular representations. Similarly, Drotár et al. [2021], Liu et al. [2022] explored autoregressive models for molecular generation, generating atoms sequentially. These models incorporate angles during generation to improve molecular detail and accuracy.

Diffusion models enable pocket-free and pocket-based drug design.Kingma et al. [2021] Hoogeboom et al. [2022] introduced Equivariant Diffusion Models (EDMs), which simultaneously learn continuous coordinates and atom types for molecule generation. Multiple studies built on this approach: GeoDiff Xu et al. [2022] predicts a molecule's 3D conformation and DiffLinkerIgashov et al. [2022] learns to connect seed fragments. Similarly, Schneuing et al. Schneuing et al. [2022] developed DiffSBDD, a denoising diffusion model for pocket-based molecule design. Guan et al's TargetDiff uses SE(3)-equivariant networks to explicitly learn the generative process for continuous coordinates and categorical atom types.Guan et al. [2023]. Peng et al introduced FragDiff Peng et al. [2022a], an autoregressive diffusion model on molecular fragments. By comparison, our approach employs order-agnostic autoregressive diffusion models Hoogeboom et al. [2021], and its high molecule validity from Geometric Vector Perceptrons Jing et al. [2020] eliminates the need for a discriminator.

3 Methods

3.1 Problem Definition

We represent the protein pocket and the ligand as point clouds with atomic coordinates r and corresponding feature vectors h. The feature vector is the one-hot encoded atom type for ligand atoms and element type, plus amino acid type for the pocket atoms. For the pocket $P = (r_i^P, h_i^P)_{i=1}^{N_P}$, and for the molecule $M = (r_i^M, h_i^M)_{i=1}^{N_M}$ where N_P and N_M are the number of atoms in the pocket and molecule respectively. We further separate each molecule into multiple fragments and molecular scaffolds $M = [(r_i^{M_F}, r_i^F), (h_i^{M_F}, h_i^F)]_{i=1}^{N_M}$. M_F and F superscripts represent molecule scaffold and the fragment respectively. Note that for each molecule, there exist multiple fragments and scaffolds.

The autoregressive diffusion process aims to generate a new fragment conditioned on a molecular scaffold and protein pocket at each step.

3.2 Diffusion Process

The diffusion process iteratively adds noise to data point x and trains a neural network to remove noise progressively (Figure 1). Generative denoising inverts the trajectory when x is unknown. This process for fragment F is conditioned on the molecular scaffold M_F and the protein pocket P:

$$p\left(z_{t-1}^F | z_t^F, M_F, P\right) = q\left(z_{t-1}^F | \hat{x}^F, z_t^F\right) \tag{1}$$

where $\hat{x} = (1/\alpha_t)z_t - (\sigma_t/\alpha_t)\hat{\epsilon}_t$ is the approximation of x^F computed by neural network ϕ using $\hat{\epsilon}_t = \phi(z_t, t, M_F, P)$. We use Geometric Vector Perceptrons (GVP) to parameterize ϕ because they outperform equivariant neural networks. Satorras et al. [2021] Following DiffLinker Igashov et al. [2022], Jing et al. [2020], Torge et al. [2023], we define the "anchor point" as the scaffold atom bonded to the fragment F. We ensure the GNN is translationally invariant by first centering the data around the anchor point a and then sampling from $\mathcal{N}(0, I)$ instead of sampling the initial noise from $\mathcal{N}(f(a), I)$ where $f(a)$ is the anchor point center of mass.

During training, we only add noise to coordinates r and feature vector h of the fragment F. We keep the scaffold molecule M_F and the protein pocket intact. The input to the neural network is the noised version of fragment z_t^F at time t and the context u, which contains the molecular scaffold M_F, the protein pocket P, and the anchor point a. The predicted noise $\hat{\epsilon}^{F_i}$ for the fragment F_i includes coordinates and feature vector $\hat{\epsilon}^{F_i} = [\hat{\epsilon}_x^{F_i}, \hat{\epsilon}_h^{F_i}]$. We only use the predicted coordinates and feature vectors for the fragment atoms and discard the rest.

Hoogeboom et al. [2021] et al. derived an objective for order agnostic diffusion models to be optimized one step at a time (see SI). Following the ARDM approach, we first sample a random ordering σ from the set of all fragment-wise molecule generation permutations S_D at each training step, where D is the number of fragments in the molecule. Next, we uniformly sample a single fragment F to reconstruct with the diffusion model. As proposed by Kingma et al. [2021], we use a simplified objective $L(t) = ||\epsilon - \hat{\epsilon}_t||^2$ that can be optimized by mini-batch gradient descent.

Additionally, we train a separate model for anchor point prediction (see SI). During sampling, this AnchorGNN (see SI) model predicts the anchor point from among the scaffold atoms. We sample the fragment size from the data distribution conditioned on the pocket size near the anchor point. We repeat the fragment generation process until we reach a maximum number of fragments or molecule size. Algorithms 1 and 2 in SI define AutoFragDiff training and sampling procedures, respectively. We compute Lennard-Jones-motivated interactions between generated fragment and pocket atoms to minimize clashes. Following a classifier-based guidance strategy Ho and Salimans [2022], we compute the gradient of a score function (Lennard-Jones interaction between pocket and fragment) with respect to fragment atom coordinates and add it to them with a negative sign (see SI).

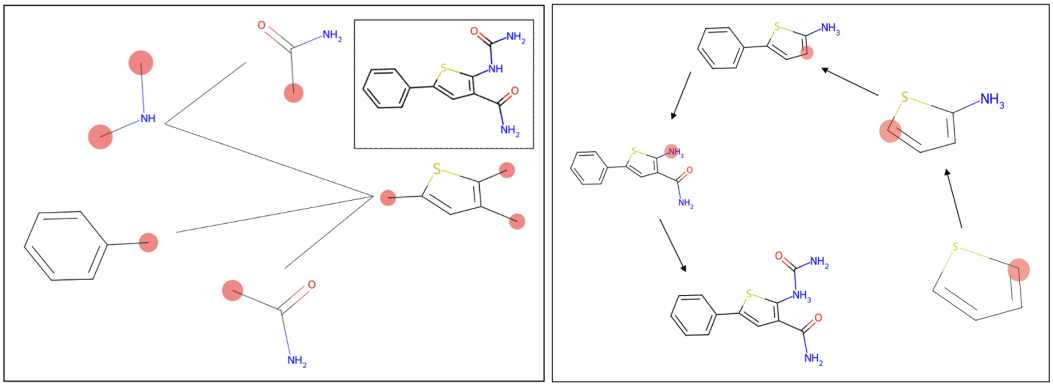


Figure 2: (**Left**) The fragment connectivity for a molecule. Highlighted atoms are the anchor points on each fragment. (**Right**) Sampled generation order for the molecule on the left from its fragments.

4 Datasets

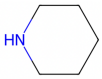
CrossDock: We use CrossDock2020Francoeur et al. [2020] to evaluate AutoFragDiff for pocket-based molecule generation. Similar to other studies, we refined the original 22.5 million docked protein binding complexes by filtering for low (<1Å) RMSD and sequence identity of less than 30%. This procedure yielded 100,000 training complexes and 100 previously unseen testing pockets. We used RDKit Landrum [2013] and BRICS Degen et al. [2008] to fragment molecules by breaking bonds between rings without breaking fused ring systems. We used a maximum of 8 fragments per molecule. We used breadth-first and depth-first search traversals of each molecule's fragment connectivity graph to avoid computing an intractable enumeration of all potential fragment-wise molecule reconstructions. At each reconstruction step, we saved the scaffold atoms and coordinates, the added fragment, and the anchor point where the scaffold connects to the next fragment. Figure 2 illustrates the molecule fragmentation strategy and generation order.

5 Results

As in TargetDiffGuan et al. [2023], we use openbabel O'Boyle et al. [2011] to reconstruct the molecules from the generated atomic point clouds. In terms of the Jensen Shannon Divergence Lin [1991] (JSD) of angles and dihedrals for common ring structures in CrossDock, AutoFragDiff significantly surpasses other models (Table 1). Although it was not a focus of this study, we also assess the generated molecules for various chemical properties (Table 2), including drug-likeliness (QED) and average synthetic accessibility (SA)Ertl and Schuffenhauer [2009]. "Diversity" evaluates the average molecular fingerprint similarity across all generated molecule pairs. AutoFragDiff generates realistic molecules with higher calculated binding affinity than the molecules in the test set and exhibits results on par with state-of-the-art models. Figure 8 and Figure 9 (see SI) show generated molecules for two examples, protein L3MBTL1 (pdb: 2pqw) and P21-activated kinase (pdb: 5i0b).

Additionally, we used PoseCheck Harris et al. [2023] to evaluate the generated molecules for clashes with protein atoms, strain energies, and interactions with pocket atoms (see SI). AutoFragDiff molecules averaged 6.7 clashes with pocket atoms (Figure 4), outperforming other diffusion-based models (TargetDiff 9.2 and DiffSBDD 11.8 averages). Non-diffusion models Pocket2Mol and 3DSBDD averaged 5.7 and 3.9 clashes per molecule, while the CrossDock ground truth test set averaged 4.8 clashes. Similarly, non-diffusion modelsPocket2Mol and 3DSBDD generally generated molecules with lower strain energies than diffusion-based models (Figure 5). Considering interaction types, TargetDiff molecules had the most H-bond donors and acceptors (Figure 6), while both AutoFragDiff and TargetDiff showed the most hydrophobic and Van der Waals interactions, on par with the CrossDock test set molecules.

Table 1: JSD of angles and dihedrals for most common rings in CrossDock dataset. Best score highlighted in dark gray; second best in light gray. DiffSBDD results are from the conditional all atoms model Schneuing et al. [2022]

								
Model	**angles**	**dihedrals**	**angles**	**dihedrals**	**angles**	**dihedrals**	**angles**	**dihedrals**
3D-SBDD	0.458	0.666	0.293	0.300	0.457	0.625	0.342	0.439
Pocket2Mol	0.438	0.574	0.321	0.272	0.347	0.551	0.408	0.478
DiffSBDD*	0.342	0.549	0.310	0.235	0.254	0.546	0.363	0.435
TargetDiff	**0.203**	**0.459**	**0.154**	**0.176**	**0.140**	**0.460**	**0.335**	**0.437**
AutoFragDiff(ours)	**0.103**	**0.151**	**0.191**	**0.172**	**0.073**	**0.179**	**0.293**	**0.333**

Scaffold Extension: Since our model adds fragments to an existing molecular scaffold at each step, it can further optimize a user-provided starting scaffold. To test the concept, we extracted the Murcko scaffold from every molecule in the CrossDock test set. We augmented each scaffold with up to 4 fragments, generating 20 distinct molecules per CrossDock molecule. 70% of the newly generated molecules exhibited higher calculated binding affinity than their corresponding starting molecule (average Vina score of -7.8 generated versus -7.1 CrossDock). Figure 3 contains representative examples for an *S. cerevisiae* Cytochrome-*c* peroxidase pocket (pdb: 1a2g).

Table 2: Pocket-based generative models comparison. Best score highlighted in dark gray; second best in light gray.

Method	Vina (↓)	Diversity (↑)	QED (↑)	SA (↑)
3D-SBDD	-6.71	0.70	**0.49**	0.62
Pocket2Mol	-7.15	0.69	**0.56**	**0.74**
DiffSBDD	-6.90	**0.73**	0.48	**0.63**
TargetDiff	**-7.55**	**0.72**	0.49	0.61
AutoFragDiff(ours)	**-7.45**	0.69	0.45	0.62
CrossDock Test Set Molecules	-7.10	-	0.47	0.73

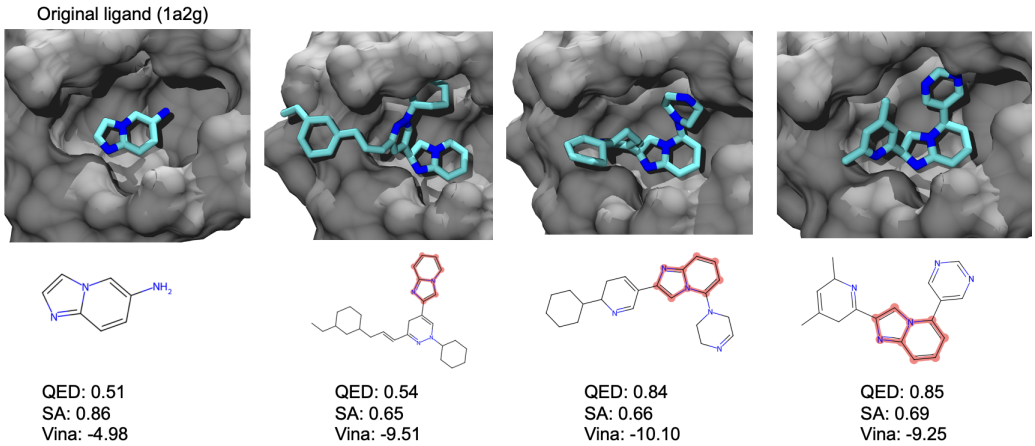


Figure 3: Scaffold (red) extension examples on a Cytochrome-*c* peroxidase (pdb: 1a2g).

6 Conclusion

We introduce AutoFragDiff (https://github.com/keiserlab/autofragdiff), an open-source autoregressive fragment-based diffusion model tailored for pocket-free and pocket-based molecule generation. A standout feature of AutoFragDiff is its capability for scaffold extension, which is a key aspect of many real-world drug design applications, especially in close-in optimization around lead series. The model is adept at generating molecules with high-quality local geometry and exhibits robust binding affinity to target proteins. Looking forward, we aim to enhance the ligand affinity within the pocket using guidance strategies and model architecture improvements.

References

Amy C Anderson. The process of structure-based drug design. *Chemistry & biology*, 10(9):787–797, 2003.

Kenneth Atz, Francesca Grisoni, and Gisbert Schneider. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12):1023–1032, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using'drug-like'chemical fragment spaces. *ChemMedChem: Chemistry Enabling Drug Discovery*, 3(10):1503–1507, 2008.

Pavol Drotár, Arian Rokkum Jamasb, Ben Day, Cătălina Cangea, and Pietro Liò. Structure-aware generation of drug-like molecules. *arXiv preprint arXiv:2111.04107*, 2021.

Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1:1–11, 2009.

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.

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. *arXiv preprint arXiv:2303.03543*, 2023.

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.

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

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

Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

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.

Ilia Igashov, Hannes Stärk, Clément Vignac, Victor Garcia Satorras, Pascal Frossard, Max Welling, Michael Bronstein, and Bruno Correia. Equivariant 3d-conditional diffusion models for molecular linker design. *arXiv preprint arXiv:2210.05274*, 2022.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael JL Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2013.

Yibo Li, Jianfeng Pei, and Luhua Lai. Structure-based de novo drug design using 3d deep generative models. *Chemical science*, 12(41):13664–13675, 2021.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

Christopher Lipinski and Andrew Hopkins. Navigating chemical space for biology and medicine. *Nature*, 432(7019):855–861, 2004.

Meng Liu, Youzhi Luo, Kanji Uchino, Koji Maruhashi, and Shuiwang Ji. Generating 3d molecules for target protein binding. *arXiv preprint arXiv:2204.09410*, 2022.

Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of cheminformatics*, 3(1):1–14, 2011.

Xingang Peng, Jiaqi Guan, Jian Peng, and Jianzhu Ma. Pocket-specific 3d molecule generation by fragment-based autoregressive diffusion models. 2022a.

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, 2022b.

Matthew Ragoza, Joshua Hochuli, Elisa Idrobo, Jocelyn Sunseri, and David Ryan Koes. Protein–ligand scoring with convolutional neural networks. *Journal of chemical information and modeling*, 57(4):942–957, 2017.

Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021.

Arne Schneuing, Yuanqi Du, Charles Harris, Arian Jamasb, Ilia Igashov, Weitao Du, Tom Blundell, Pietro Lió, Carla Gomes, Max Welling, et al. Structure-based drug design with equivariant diffusion models. *arXiv preprint arXiv:2210.13695*, 2022.

Miha Skalic, Davide Sabbadin, Boris Sattarov, Simone Sciabola, and Gianni De Fabritiis. From target to drug: generative modeling for the multimodal structure-based ligand design. *Molecular pharmaceutics*, 16(10):4282–4291, 2019.

Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

Jos Torge, Charles Harris, Simon V Mathis, and Pietro Lio. Diffhopp: A graph diffusion model for novel drug design via scaffold hopping. *arXiv preprint arXiv:2308.07416*, 2023.

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

7 Appendix

A.1 Training and Sampling

Algorithm 1 Training

Input: Fragment x^F, Scaffold M_F, anchor point a, protein pocket P, neural network ϕ
Sample Permutation order $\sigma \sim S_D$
Sample fragment F
Sample $t \sim \mathcal{U}(0, \ldots, T)$, $\quad \epsilon_t \sim \mathcal{N}(0, I)$
$z_t^F \leftarrow \alpha_t x^F + \sigma_t \epsilon_t$
$\hat{\epsilon}_t \leftarrow \phi(z_t, M_F, a, t, \mathrm{P})$
Minimize $\|\epsilon - \hat{\epsilon}_t\|_2$

Algorithm 2 Sampling

for i in $1..D$; **do**
    Input: Scaffold M_{F_i}, anchor point a_i, protein pocket P, neural network ϕ
    Center everything at $f(a_i)$
    Sample $z_T^{F_i} \sim \mathcal{N}(0, I)$
    for t in $T; T-1; \ldots; 1$ **do**
        Sample $\epsilon_t \sim \mathcal{N}(0, I)$
        $\hat{\epsilon}_t \leftarrow \phi(z_t^{F_i}, t, M_{F_i}, a_i, P)$
        $z_{t-1}^{F_i} \leftarrow (1/\bar{\alpha}_t) \cdot z_t - \bar{\sigma}_t^2/(\bar{\alpha}_t \sigma_t) \cdot \hat{\epsilon}_t + \zeta_t.\epsilon$
    end for
    Sample $x^{F_i} \sim p(x^{F_i} | z_0^{F_i}, M_{F_i}, a_i, P)$
end for

For sampling molecule sizes, we first bin the pocket volumes into 10 bins (using grids inside the protein pocket) and find the distribution of molecule sizes for each bin. During sampling, we sample molecule sizes from the distribution of the corresponding volume bin. For the first generation step, the anchor point is selected from the pocket atoms in contact with the original ligand. We first bin the pocket volume within 3.5 Å of the anchor point for fragment size and then sample the fragment sizes from the corresponding bin. The average size of generated molecules from our model is 26 atoms.

A.2 Diffusion Process

At each timestep $t = 0...T$ the conditional distribution of the intermediate state z_t^F for a single fragment F given the previous state is defined by the multivariate normal distribution:

$$q(z_t^F | z_{t-1}^F) = N\left(z_t^F; \bar{\alpha}_t z_{t-1}^F, \bar{\sigma}_t^2 I\right) \tag{2}$$

In this equation $\bar{\alpha}_t = \alpha_t / \alpha_{t-1}$ controls how much signal is retained and $\bar{\sigma}_t = \sigma_t^2 - \bar{\alpha}_t^2 \sigma_{t-1}^2$ controls how much noise is added. The full transition model for diffusion is Markovian:

$$q(z_0^F, z_1^F, ..., z_T^F | x^F) = q(z_0^F | x^F) \prod_{t=1}^{T} q(z_t^F | z_{t-1}^F) \tag{3}$$

The true denoising process has a closed-form solution when conditioned on x^F:

$$q(z_{t-1}^F | z_t^F, x^F) = N\left(z_{t-1}^F; \mu(x, z_t), \zeta_t^2 I\right) \tag{4}$$

where $\mu_t(x^F, z_t^F)$ and ζ_t have analytical solutions:

$$\mu_t(x^F, z_t^F) = \frac{\bar{\alpha}_t \sigma_{t-1}^2}{\sigma_t^2} z_t + \frac{\alpha_s \bar{\sigma}_t^2}{\sigma_t^2}, \quad \zeta_t = \frac{\bar{\sigma}_t \sigma_{t-1}}{\sigma_t} \tag{5}$$

We trained AutoFragDiff with $T = 500$ diffusion steps using a polynomial noise scheduler:

$$\alpha_t = (1 - 2s) \cdot (1 - (t/T)^2) \tag{6}$$

where $s = 10^{-5}$ is the precision value to help with numerical issues.

A.3 Geometric Vector Perceptrons

GVP Jing et al. [2020] uses nodes with scalar features s as inputs. These scalars represent embedded features of atoms without accompanying vector features. Edges within the graph incorporate a normed direction vector alongside the distance between two nodes. More specifics about this can be found in GVP paper.Jing et al. [2020]

As described previously in DiffHopp Torge et al. [2023], the attributes of nodes and edges undergo linear transformations. Edge embeddings are achieved in two phases: initially, their inputs are normalized using layer normalization Ba et al. [2016], and following this, they are channeled through a GVP. Here, both σ and σ^+ operate as the identity function, resulting in a scalar with a hidden size of $h/2$ and a singular vector. Nodes undergo a parallel embedding process, culminating in outputs of h scalars and $h/2$ vectors, summing up to h features. The message-passing layers can be expressed as:

$$\begin{aligned} m'_{vw} &= \phi_e(h_v, h_w, e_{vw}), \\ m'_v &= \sum_{w \in N_v} \tilde{e}_{vw} m_{vw} \\ h'_v &= \phi_h(h_v, m'_v) \end{aligned} \tag{7}$$

Within this equation, $\tilde{e}_{vw} = \phi_{att}(m_{vw})$ acts as an attention mechanism, enabling the learning of soft edge estimates, mirroring the approach in EGNN. The function ϕ_e combines three GVPs featuring hidden sizes $(h, h/2)$. Notably, the final GVP has σ as its identity function. Meanwhile, ϕ_{att} embodies a single GVP translating to a singular scalar with σ functioning as the sigmoid activation. A factor of $C = 100$ normalizes the resulting output.

The relationship between $\phi_h(h_v, m'_v)$ is captured by the equation $\phi_h(h_v, m'_v) = \text{norm}(h_v + \phi'_h(\text{norm}(h_v + m'_v)))$. This employs a residual architecture where ϕ'_h integrates two GVPs with sizes $(h, h/2)$. This encapsulates input, hidden, and output dimensions. The terminal layer once again adopts σ as the identity function. The term "norm" represents layer normalization, which isn't applied to vectors.

A.4 Autoregressive Diffusion Models

Autoregressive models can factorize a multivariate distribution into a product of D univariate distributions.

$$\log p(x) = \sum_{t=1}^{D} \log p(x_t | x_1, ..., x_{t-1}) \tag{8}$$

Sampling from such models can be done through D iterative sampling steps. Order agnostic models can generate variables with random orderings $\sigma \in S_D$ where S_D is the set of all permutations for building the molecule from its fragments. The log-likelihood of these models can be written as:

$$\log p(M|P) \geq \mathbb{E}_{\sigma \sim \mathcal{U}(S_D)} \sum_{i=1}^{D} \log p(M_{\sigma_i} | M_{\sigma(<i)}, P) \tag{9}$$

In this equation, M is the set of molecule atoms, P is the set of protein atoms, and M_{σ_i} is the molecule generated with sampled ordering σ at the fragment step i. Hoogeboom et. al. derived

an objective for order agnostic diffusion models Hoogeboom et al. [2021] that only needs to be optimized for a single step at a time:

$$\log p(M|P) \geq \mathbb{E}_{\sigma \sim U(S_D)} D.\mathbb{E}_{i \sim \mathcal{U}(1..D)} \log p(M_{\sigma_i}|M_{\sigma(<i)}, P) \tag{10}$$

According to this objective, during training, we sample a random order σ of molecule generation uniformly from the set of all generation orders S_D, and a single fragment from the uniform distribution of all fragments in the molecule. We train the diffusion model to predict this single fragment. In practice, we optimize a simplified $L(t) = ||\epsilon - \hat{\epsilon}_t||^2$ loss by mini-batch gradient descent.

A.5 Hyperparameters

We consider the protein graph as the protein atoms within 7 Å of the original ligand. Edges within the ligand are fully connected, while protein-ligand and protein-protein edges are drawn with a radius threshold of 4.5 Å. The edge features for nodes i and j are the distance d_{ij} and the normalized direction vector $(x_i - x_j)/d_{ij}$. As previously suggested by Hoogeboom et al. [2022], we scale node types h by a factor of 0.25. The final model had 6 GVP layers, with hidden dimension of 128 and a joint embedding dimension of 32. We trained the model with a learning rate of 2×10^{-4} for 500 epochs.

A.6 Additional results

Table 3 compares different models by JSD for different types of bonds.

Table 3: Comparison of JSD of bond distances in different bond types for different models. Best results per row are highlighted in dark gray, and second best in light gray.

Bond	3D-SBDD	Pocket2Mol	DiffSBDD	TargetDiff	AutoFragDiff
C-C	0.576	0.455	0.347	**0.286**	**0.363**
C=C	0.421	0.561	0.314	**0.220**	**0.221**
C-N	0.383	0.321	0.313	**0.242**	**0.290**
C=N	0.443	0.377	0.348	**0.179**	**0.231**
C-O	0.394	0.326	0.353	**0.298**	**0.354**
C=O	0.511	0.446	0.398	**0.398**	**0.363**
C:C	0.459	0.309	0.316	**0.176**	**0.295**
C:N	0.582	0.377	0.348	**0.158**	**0.217**

As described below, we used PoseCheck Harris et al. [2023] to evaluate each model's generated molecules for their number of clashes with pocket atoms, strain energies, and interactions (hydrogen bond donors and acceptors, hydrophobic interactions, and Van der Waals contacts) with pocket atoms.

7.1 Steric clashes

We computed steric clashes of generated molecules with protein atoms with a clash tolerance of 0.5 Å, as described in PoseCheck. We use a classifier guidance approach to minimize the clashes to pocket atoms in our model. We calculate the Lennard-Jones (LJ) interaction of the fragment atoms with pocket atoms as the guidance function and add the negative gradient of this score with respect to fragment coordinates to the current fragment atom coordinates. When computing the Lennard-Jones interactions, we include pocket hydrogen atoms, although the ligand diffusion itself does not include explicit hydrogens.

$$U(r) = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{11}$$

In this equation, σ is the sum of Van der Waals radii of the pocket and ligand atoms, and r is the distance between protein and ligand atoms. We clip the output at 1000 to avoid very large values.

$$x^F = x^F - \epsilon \nabla_{x^F} U(r) \tag{12}$$

We use a cosine-β weight scheduler to progressively lower the effect of LJ guidance over the inference trajectory. Note that the LJ guidance is only used for avoiding clashes with pocket atoms and does not have a physical meaning. CrossDock test set molecules have on average 4.8 clashes with pocket atoms. Our model shows an average of 6.7 clashes, outperforming other diffusion models Targetdiff: 9.2 and DiffSBDD: 11.8. However, non-diffusion-based models have fewer clashes with pocket atoms, with Pocket2Mol having an average of 5.7 and 3DSBDD 3.9 clashes.

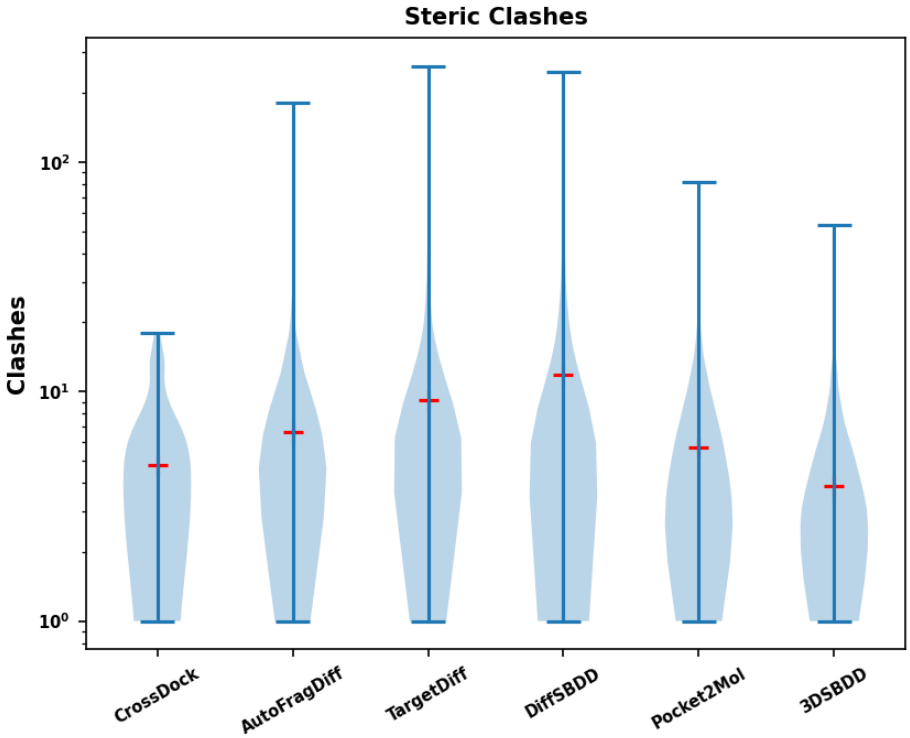


Figure 4: Steric clashes of different models

7.2 Strain energy

We computed molecule strain energies as the difference between the internal energy of the generated and minimized conformers, using the Universal Force Field (UFF) from RDKit (Figure 5). Diffusion-based models all showed higher strain energies than non-diffusion models with Pocke2Mol model having the lowest strain energy.

7.3 Ligand-pocket interactions

Using PoseCheck, we computed four interaction types between molecule poses and pocket atoms (hydrogen bond donors, hydrogen bond acceptors, hydrophobic interactions, and Van der Waals contacts) (Figure 6). CrossDock molecules have more hydrogen bond donors than any generated molecules. TargetDiff shows more H-bond donors and acceptors than the other generative models. In terms of hydrophobic and Van der Waals interactions, AutoFragDiff and TargetDiff show similar performance which is also on par with CrossDock test set molecules

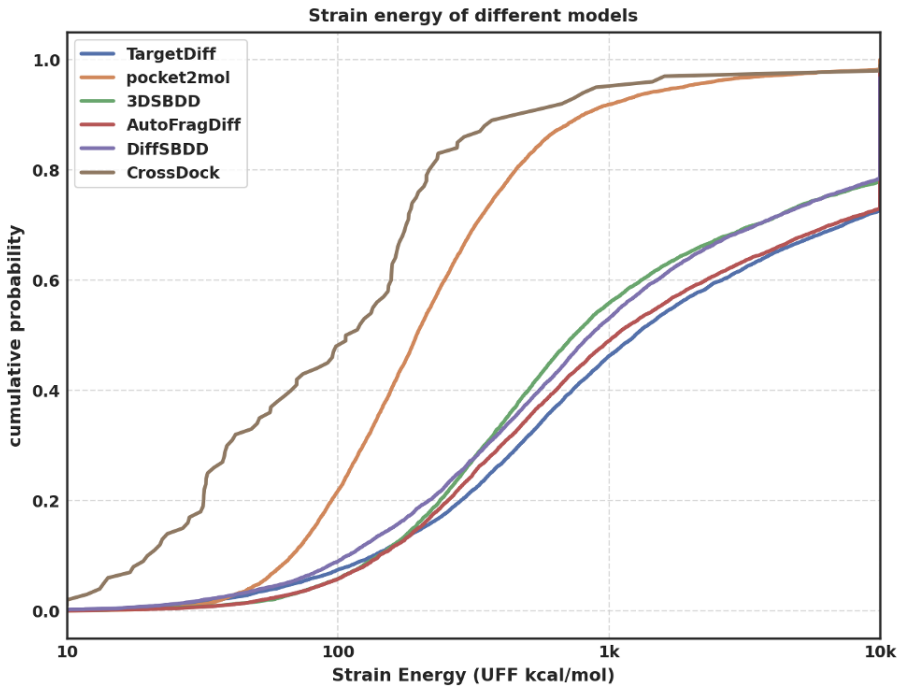


Figure 5: Strain energies of different models

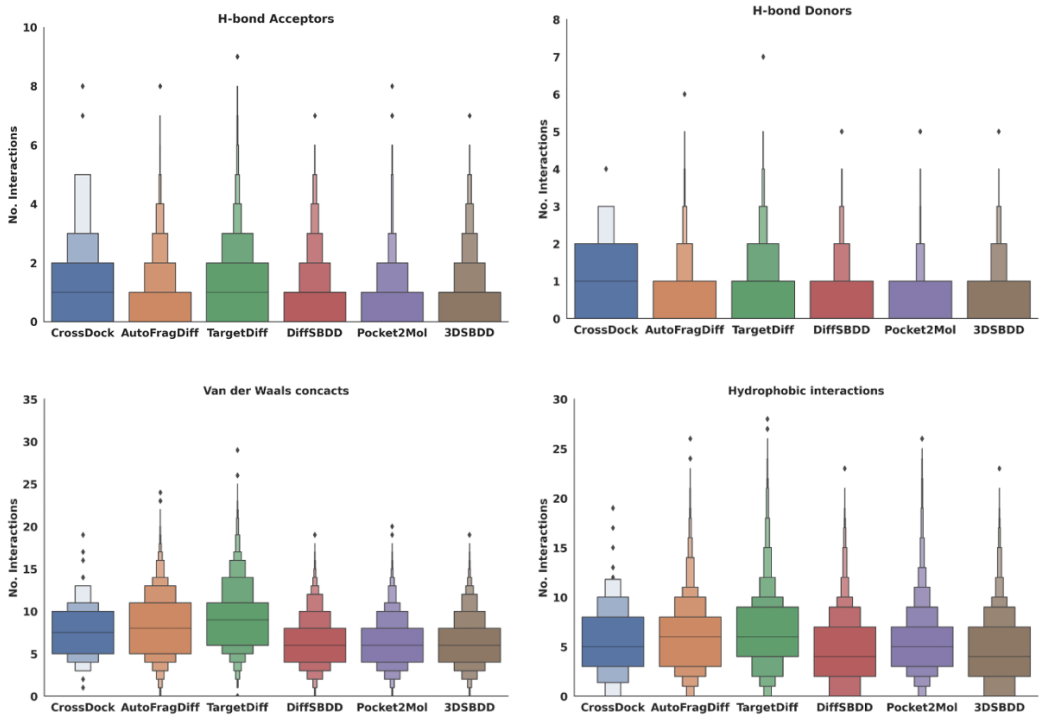


Figure 6: Different interaction types of generated poses with protein atoms.

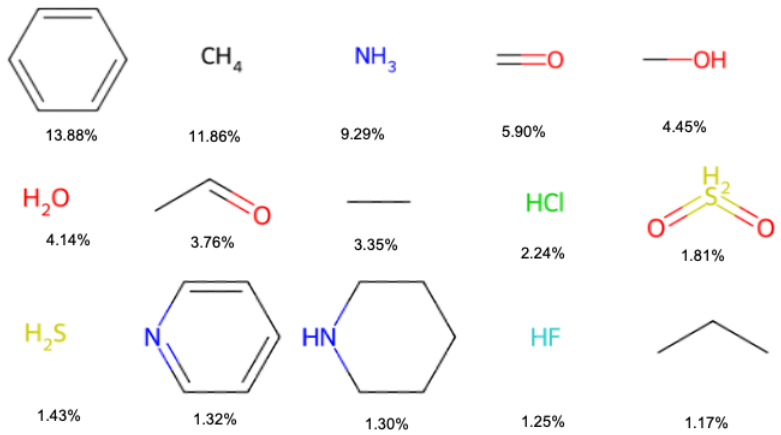


Figure 7: Top 15 occurring fragments using our custom fragmentation in CrossDock dataset.

A.7 Fragmentation

During fragmentation, we first break all the bonds between rings without breaking fused ring systems. In addition, we also use RDKit and use BRICS to fragment the molecules (Figure 7). A maximum of 8 fragments is used for each molecule; if the number of fragments exceeds 8, we connect the smallest fragments iteratively until the maximum of 8 fragments is reached. Given the fragment connectivity of a molecule (fragment adjacency), we compute all BFS and DFS traversals of the molecule graph based on the fragment. We only use BFS And DFS traversals to avoid computing all of the molecule's fragmentation graph's traversals for computational feasibility. A generation order defines how fragments are added step-wise based on their connectivity to make a complete molecule. At each step of the generation order, we save the scaffold atoms and their coordinates, the added fragment at the generation step and its coordinates, the generation step, and the scaffold anchor point for the next fragment. This strategy greatly augments the size of the dataset as well.

A.8 Anchor point predictor

We trained a standalone neural network (AnchorGNN) to predict the anchor points during sampling. We used graph convolutional layers (GCL) to predict the probability of each scaffold atom being an anchor point. The molecular scaffold and protein pocket atoms are the model inputs. We use one-hot encoded atom types as scaffold atom features. For pocket atoms, the features are atom types, amino acid types, and whether the atom belongs to the backbone or sidechain. We use inter-atomic squared distance $d_{ij}^2 = ||r_i - r_j||^2$ as the edge feature.

The update for feature h and coordinates of node i at layer l are computed as follows:

$$m_{ij} = \phi_e(h_i^l, h_j^l, d_{ij}^2), \quad h_i^{l+1} = \phi_h(h_i^l, \sum_{j \neq i} m_{ij}), \quad r_i^{l+1} = r_i^l + \phi_{vel}(r^l, h^l, i) \tag{13}$$

with $d_{ij} = ||r_i - r_j||$ and ϕ_e, ϕ_h being learning functions. We perform a sequence of l Graph Convolutional Layers ($l = 4$). Finally, node embeddings for the scaffold molecule h^M are linearly transformed to a single number and passed through a sigmoid function to compute the probabilities. A binary cross-entropy loss is used to train the model. During sampling, we take the anchor point with the highest probability. A learning rate of 5×10^{-4} was used to train this model.

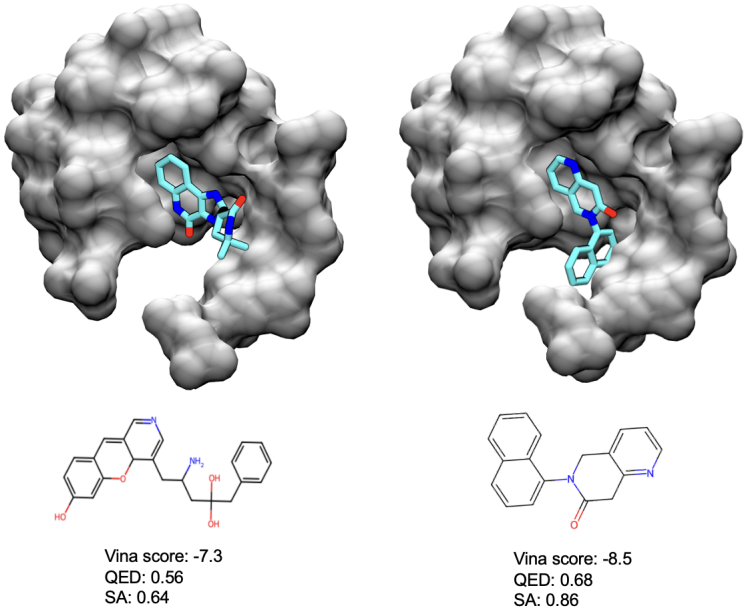


Figure 8: Examples of generated molecules for protein L3MBTL1 (pdb: 2pqw).

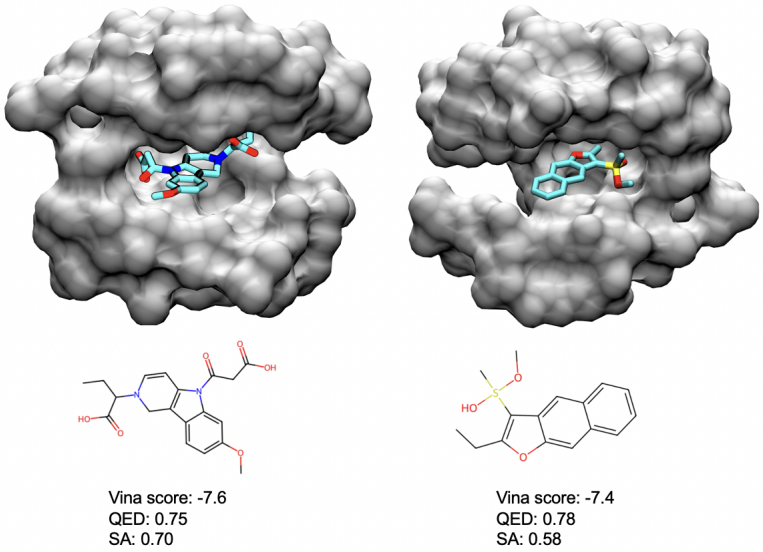


Figure 9: Examples of generated molecules for P21-activated kinase (pdb: 5i0b).