



Equivariant 3D-conditional diffusion model for molecular linker design

In the format provided by the authors and unedited

CONTENTS

1	Background	2
A	Applications	2
B	Related work	2
2	Equivariance	4
A	Proof of Proposition 1	4
B	Problem with translations	4
3	Implementation details	6
A	Dynamics	6
B	SizeGNN	6
C	Training	7
4	Evaluation methodology	8
A	Evaluation details	8
B	2D filters	9
C	DeLinker and 3DLinker on GEOM Dataset	9
D	GNINA for Hsp90 docking	10
5	Additional results	11
A	GEOM	11
B	Pockets	11
C	Inpainting	11

1. BACKGROUND

A. Applications

There are several drug discovery strategies in which one of highly relevant tasks is to design linkers for molecular fragments that are placed in the space and have fixed positions.

Fragment Based Drug Discovery (FBDD) By analogy with classical drug discovery methods, one of the common strategies in FBDD is to operate on fragments that interact with the target proteins. First, binding fragments are defined and characterized (using high-throughput screening followed by X-ray / NMR, or virtual screening and docking). As a result, the exact location and orientation of the fragments in which they interact with patches of the protein pocket is defined. The next step is to design a linker between the fragments that preserves positions and thus the binding affinity of the fragments (preferably, the linker addition will enhance the binding affinity of the final molecule) [1]. Several successes have been reported on the design of linkers starting from a protein crystal structure in complex with bound fragments [1], for example: inhibitors for CK2 [2], LDH-A [3] and Dot1L [4], which are proteins playing crucial roles in cancer, were designed by linking the fragments that were experimentally observed in a bound state with the corresponding targets.

Proteolysis targeting chimera (PROTAC) PROTACs are heterobifunctional small molecules designed for stimulating degradation of a target protein by bringing it to the proximity of an E3-ligase. PROTACs consist of two ligands joined by a linker: one ligand recruits and binds a target protein while the other recruits and binds the E3 ubiquitin ligase [5]. For designing PROTACs, one of possible strategy is to dock two proteins with ligands bound and explore a favorable conformation of the prospective tertiary complex. This information about the initial docking pose of the proteins and exact positions of bound fragments is further used for designing linkers that stabilize the whole complex [6, 7].

Scaffold hopping Scaffold hopping is a strategy for designing novel compounds by replacing the central core structure of the known molecule. As shown by Sun et al. [8], various scaffold-hopping strategies rely on the experimental 3D data of the initial compound bound to a target: the information about the geometry of the initial bound molecule is important for altering its core with the increase of the binding affinity, potency or selectivity of the whole molecule. In such a case, scaffold-hopping of the bound molecule can be considered as a linking problem of several disconnected fragments with fixed known 3D coordinates.

B. Related work

Molecular linker design has been widely used in the fragment-based drug discovery community [9]. Various *de novo* design methods refer to the fragment linking problem [10–16]. Early fragment linking methods were based on search in predefined libraries of linkers [17, 18], genetic algorithms, tabu search [19] and force field optimization [20]. Having been successfully used in multiple application cases [21–23], these methods are however computationally expensive and substantially limited by the available data.

Hence, there has recently been interest in developing learning-based methods for molecular linker design. Yang et al. [24] proposed SyntaLinker, a SMILES-based deep conditional transformer neural network that solves a sentence completion problem [25]. This method inherits the drawbacks of SMILES, which are absent of 3D structure and the lack of consistency where atoms that are close in the molecule can be far away in the SMILES string. Imrie et al. [26] overcome these limitations by introducing an autoregressive model DeLinker, and its extension DEVELOP [27] that uses additional pharmacophore information. Although these methods operate on 3D molecular conformations, they use very limited geometric information and require input on the attachment atoms of the fragments. Recently, Huang et al. [28] proposed another autoregressive method, 3DLinker, that does not require one to specify attachment points and leverages the geometric information to a much greater extent. It makes this approach more relevant for connecting docked fragments. As both DeLinker and 3DLinker are autoregressive models, they are not permutation equivariant which limits their sample efficiency and ability to scale to large molecules [29, 30]. Moreover, these methods are capable of connecting only pairs of fragments and cannot be easily extended to larger sets of fragments.

Beyond the linker design problem, several groups have recently proposed diffusion models for molecule generation [31, 32] and molecular conformer generation [33–35]. Several others apply diffusion models for docking [36], structure-based drug design [37–39], protein design [40–43], and protein-ligand complex prediction [44]. Some of these methods generate molecules conditioned on additional 3D contexts such as protein pockets and binding sites, antibody complementarity-determining regions (CDR), and protein backbones. In many cases, conditioning can be achieved by training an unconditional diffusion model on the full molecules (including context) and replacing the learned denoising process with the true denoising process for the known context during sampling. Such a mechanism was originally proposed by Sohl-Dickstein et al. [45] and later improved by Lugmayr et al. [46] in the scope of the image inpainting

problem. We discuss the limitations of this approach in the context of the molecular linker design problem and provide quantitative results in Section 5C. In this work, we propose another 3D-conditioning approach and show that it satisfies the desirable equivariance properties.

2. EQUIVARIANCE

A. Proof of Proposition 1

O(3)-equivariance of function f and the fact that q is isotropic Gaussian distribution implies O(3)-equivariance of the prior distribution:

$$p(\mathbf{R}z_T|\mathbf{R}\mathbf{u}) = \mathcal{N}(\mathbf{R}z_T|f(\mathbf{R}\mathbf{u}), \mathbf{I}) = \mathcal{N}(\mathbf{R}z_T|\mathbf{R}f(\mathbf{u})) = \mathcal{N}(z_T|f(\mathbf{u})) = p(z_T|\mathbf{u}).$$

Likewise, O(3)-equivariance of function φ and Equation 8 imply O(3)-equivariance of all transition probabilities $p(z_{t-1}|z_t, \mathbf{u})$.

To obtain the distribution $p(z_0|\mathbf{u})$ of data point z_0 , we can consider joint distribution $p(z_0, z_1, \dots, z_T|\mathbf{u})$ and marginalize it by $z_{1..T}$:

$$p(z_0|\mathbf{u}) = \int p(z_0, z_1, \dots, z_T|\mathbf{u}) dz_{1..T} = \int p(z_T|\mathbf{u}) \prod_{t=0}^{T-1} p(z_t|z_{t+1}, \mathbf{u}) dz_{1..T}.$$

Having prior and all transition distributions equivariant, it is now trivial to show O(3)-equivariance of $p(z_0|\mathbf{u})$:

$$\begin{aligned} p(\mathbf{R}z_0|\mathbf{R}\mathbf{u}) &= \int p(\mathbf{R}z_T|\mathbf{R}\mathbf{u}) \prod_{t=0}^{T-1} p(\mathbf{R}z_t|\mathbf{R}z_{t+1}, \mathbf{R}\mathbf{u}) dz_{1..T} \\ &= \int p(z_T|\mathbf{u}) \prod_{t=0}^{T-1} p(z_t|z_{t+1}, \mathbf{u}) dz_{1..T} \quad (\text{equivariant prior } p(z_T|\mathbf{u})) \\ &= \int p(z_T|\mathbf{u}) \prod_{t=0}^{T-1} p(z_t|z_{t+1}, \mathbf{u}) dz_{1..T} \quad (\text{equivariant kernels } p(z_t|z_{t+1}, \mathbf{u})) \\ &= \int p(z_0, z_1, \dots, z_T|\mathbf{u}) dz_{1..T} = p(z_0|\mathbf{u}). \end{aligned}$$

B. Problem with translations

Consider transition probability $p(z_{t-1}|z_t, \mathbf{u}) = q(z_{t-1}|\hat{\mathbf{x}}, z_t)$. Translation equivariance of $p(z_{t-1}|z_t, \mathbf{u})$ means that

$$p(z_{t-1} + \mathbf{t}|z_t + \mathbf{t}, \mathbf{u} + \mathbf{t}) = p(z_{t-1}|z_t, \mathbf{u}) \quad \forall \mathbf{t} \in \mathbb{R}^3. \quad (\text{S1})$$

More precisely,

$$\mathcal{N}(z_{t-1} + \mathbf{t}; \hat{\boldsymbol{\mu}}_t(z_t + \mathbf{t}, \mathbf{u} + \mathbf{t}), \zeta_t^2 \mathbf{I}) = \mathcal{N}(z_{t-1}; \hat{\boldsymbol{\mu}}_t(z_t, \mathbf{u}), \zeta_t^2 \mathbf{I}), \quad (\text{S2})$$

where

$$\hat{\boldsymbol{\mu}}_t(z_t, \mathbf{u}) = \boldsymbol{\mu}_t(\hat{\mathbf{x}}, z_t) = \frac{\bar{\alpha}_t \sigma_t^2}{\sigma_t^2} z_t + \frac{\alpha_{t-1} \bar{\sigma}_t^2}{\sigma_t^2} \hat{\mathbf{x}}, \quad (\text{S3})$$

and

$$\hat{\mathbf{x}} = \frac{1}{\alpha_t} z_t - \frac{\sigma_t}{\alpha_t} \varphi(z_t, \mathbf{u}, t). \quad (\text{S4})$$

Therefore, the mean of this distribution can be written as:

$$\hat{\boldsymbol{\mu}}_t(z_t, \mathbf{u}) = \frac{1}{\alpha_t} z_t - \frac{\bar{\sigma}_t^2}{\alpha_t \sigma_t} \varphi(z_t, \mathbf{u}, t). \quad (\text{S5})$$

Neural network φ is translation invariant meaning that $\varphi(z_t + \mathbf{t}, \mathbf{u} + \mathbf{t}, t) = \varphi(z_t, \mathbf{u}, t)$. It means that:

$$\hat{\boldsymbol{\mu}}_t(z_t + \mathbf{t}, \mathbf{u} + \mathbf{t}) = \frac{1}{\alpha_t} (z_t + \mathbf{t}) - \frac{\bar{\sigma}_t^2}{\alpha_t \sigma_t} \varphi(z_t + \mathbf{t}, \mathbf{u} + \mathbf{t}, t) \quad (\text{S6})$$

$$= \frac{1}{\alpha_t} (z_t + \mathbf{t}) - \frac{\bar{\sigma}_t^2}{\alpha_t \sigma_t} \varphi(z_t, \mathbf{u}, t) \quad (\text{S7})$$

$$= \frac{1}{\alpha_t} z_t - \frac{\bar{\sigma}_t^2}{\alpha_t \sigma_t} \varphi(z_t, \mathbf{u}, t) + \frac{1}{\alpha_t} \mathbf{t} \quad (\text{S8})$$

$$= \hat{\boldsymbol{\mu}}_t(z_t, \mathbf{u}) + \frac{1}{\alpha_t} \mathbf{t} \quad (\text{S9})$$

$$= \hat{\boldsymbol{\mu}}_t(z_t, \mathbf{u}) + \lambda_t \mathbf{t}. \quad (\text{S10})$$

So we see that $\hat{\mu}_t(\mathbf{z}_t, \mathbf{u})$ is equivariant to translations, however input and output translations **are not equal** because $\lambda \neq 1$. It means that equivariance of distributions from Equations (S1) and (S2) does not hold. More formally,

$$p(\mathbf{z}_{t-1} + \mathbf{t} | \mathbf{z}_t + \mathbf{t}, \mathbf{u} + \mathbf{t}) = \mathcal{N}(\mathbf{z}_{t-1} + \mathbf{t}; \hat{\mu}_t(\mathbf{z}_t, \mathbf{u}) + \lambda \mathbf{t}, \zeta_t^2 \mathbf{I}) \quad (\text{S11})$$

$$= \mathcal{N}(\mathbf{z}_{t-1}; \hat{\mu}_t(\mathbf{z}_t, \mathbf{u}) + (\lambda - 1)\mathbf{t}, \zeta_t^2 \mathbf{I}) \quad (\text{S12})$$

$$\neq \mathcal{N}(\mathbf{z}_{t-1}; \hat{\mu}_t(\mathbf{z}_t, \mathbf{u}), \zeta_t^2 \mathbf{I}). \quad (\text{S13})$$

We can also write that

$$p(\mathbf{z}_{t-1} + \mathbf{t} | \mathbf{z}_t + \mathbf{t}, \mathbf{u} + \mathbf{t}) = p(\mathbf{z}_{t-1} + (1 - \lambda)\mathbf{t} | \mathbf{z}_t, \mathbf{u}) \neq p(\mathbf{z}_{t-1} | \mathbf{z}_t, \mathbf{u}). \quad (\text{S14})$$

3. IMPLEMENTATION DETAILS

Algorithm S1. Training

Input: linker x , context u , neural network φ
 Sample $t \sim \mathcal{U}(0, \dots, T)$, $\epsilon_t \sim \mathcal{N}(0, I)$
 $z_t \leftarrow \alpha_t x + \sigma_t \epsilon_t$
 $\hat{\epsilon}_t \leftarrow \varphi(z_t, u, t)$
 Minimize $\|\epsilon_t - \hat{\epsilon}_t\|^2$

Algorithm S2. Sampling

Input: context u , neural network φ
 Center context u at $f(u)$
 Sample $z_T \sim \mathcal{N}(0, I)$
for t in $T, T-1, \dots, 1$:
 Sample $\epsilon_t \sim \mathcal{N}(0, I)$
 $\hat{\epsilon}_t \leftarrow \varphi(z_t, u, t)$
 $z_{t-1} \leftarrow (1/\bar{\alpha}_t) \cdot z_t - \bar{\sigma}_t^2 / (\bar{\alpha}_t \sigma_t) \cdot \hat{\epsilon}_t + \zeta_t \cdot \epsilon_t$
end for
 Sample $x \sim p(x|z_0, u)$

A. Dynamics

EGNN takes as input a graph of atoms belonging to the linker z_t and its context u represented by feature vectors $h_i \in \mathbb{R}^{\text{in}}$ and coordinates $r_i \in \mathbb{R}^3$. Feature vector h_i consists of atom types, fragments flag and time step t . If anchors are known, additionally anchor flag is passed. If the model is conditioned on the protein pocket, an additional pocket flag is passed.

Forward pass

First, atom features are passed to the encoder: $h_i \rightarrow \text{Linear}(\text{in}, \text{nf}) \rightarrow h_i^0$.
 Next, L Equivariant Graph Convolutional Layers (EGCL) are sequentially applied. Learnable components of EGCL ϕ_e, ϕ_h, ϕ_r are implemented as neural networks that include fully-connected layers (FC), batch normalization layers (BN) and activations SiLU.

Message ϕ_e

Takes a pair of node embeddings h_i^l and h_j^l and the squared distance $d_{ij}^2 = \|r_i - r_j\|^2$ between these nodes and outputs a message $m_{ij} \in \mathbb{R}^{\text{nf}}$:

$$\text{concat}[h_i^l, h_j^l, d_{ij}^2] \rightarrow \{\text{FC}(2 \cdot \text{nf} + 1, \text{nf}) \rightarrow \text{SiLU} \rightarrow \text{FC}(\text{nf}, \text{nf}) \rightarrow \text{SiLU}\} \rightarrow m_{ij}$$

Features update ϕ_h

Takes as input node embedding h_i^l and its aggregated message $m_i = \sum_j m_{ij}$ and returns the updated node embedding:

$$\begin{aligned} & \text{concat}[h_i^l, m_i] \rightarrow \\ & \rightarrow \{\text{FC}(2 \cdot \text{nf}, \text{nf}) \rightarrow \text{BN} \rightarrow \text{SiLU} \rightarrow \text{FC}(\text{nf}, \text{nf}) \rightarrow \text{BN} \rightarrow \text{add}(h_i^l)\} \rightarrow \\ & \rightarrow h_i^{l+1} \end{aligned}$$

Coordinates update ϕ_r

Takes the same input as ϕ_e and outputs a scalar value:

$$\begin{aligned} & \text{concat}[h_i^l, h_j^l, d_{ij}^2] \rightarrow \\ & \rightarrow \{\text{FC}(2 \cdot \text{nf} + 1, \text{nf}) \rightarrow \text{SiLU} \rightarrow \text{FC}(\text{nf}, \text{nf}) \rightarrow \text{SiLU} \rightarrow \text{FC}(\text{nf}, 1)\} \rightarrow \\ & \rightarrow \text{output} \end{aligned}$$

B. SizeGNN

Graph neural network for predicting probabilities of the number of atoms in the prospective linker for a given set of fragments takes as input a fully-connected graph of atoms belonging to the fragments represented by feature vectors

Table S1. Hyper parameters of EGNN models trained on ZINC, Multi-Fragment and Pocket datasets.

dataset	given anchors	nf	L	batch size	epochs	time per epoch, min
ZINC	no	128	8	128	300	15.2
ZINC	yes	128	8	128	300	17.1
Multi-Fragment	no	128	6	64	839	10.6
Multi-Fragment	yes	128	6	128	1240	10.1
Pocket (full atomic representation)	yes	128	6	32	420	20.3
Pocket (backbone representation)	yes	128	6	32	620	10.7
Pocket (no pocket)	yes	128	6	32	670	8.4

Table S2. Hyper parameters of SizeGNN models trained on ZINC and Multi-Fragment datasets.

dataset	in	hid	out	L	batch size	epochs	time per epoch, min
ZINC	8	256	10	5	256	53	10.6
Multi-Fragment	9	256	33	5	256	119	9.2

$h_i \in \mathbb{R}^{\text{in}}$ and inter-atomic squared distances $d_{ij}^2 = \|r_i - r_j\|^2$, and outputs a vector of probabilities corresponding to the predefined linker sizes $p \in [0, 1]^{\text{out}}$.

Forward pass

First, node embeddings are computed: $h_i \rightarrow \text{Linear}(\text{in}, \text{nf}) \rightarrow h_i^0$.

Next, a sequence of L Graph Convolutional Layers (GCL) is applied. Learnable components of GCL ϕ_e, ϕ_h are implemented in the same way as for EGNN. Finally, node embeddings h_i^L are projected onto \mathbb{R}^{out} , aggregated and normalized resulting in the vector of label probabilities:

$$h_i^L \rightarrow \{\text{FC}(\text{nf}, \text{out}) \rightarrow \text{Mean} \rightarrow \text{Softmax}\} \rightarrow p$$

C. Training

We trained all DiffLinker models with $T = 500$ diffusion steps using polynomial noise schedule:

$$\alpha_t = (1 - 2s) \cdot (1 - (t/T)^2), \tag{S15}$$

where $s = 10^{-5}$ is a precision value that helps to avoid numerically unstable situations [31]. We trained separate models for ZINC, Multi-Frag and Pocket datasets. Hyper parameters of the models and average time required for training one epoch are provided in Table S1. All models were trained on a single Tesla V100-PCIE-32GB GPU using Adam with learning rate $2 \cdot 10^{-5}$ and weight decay 10^{-13} .

We trained two SizeGNN models — for ZINC and Multi-Frag datasets. Hyper parameters of the models and average time required for training one epoch are provided in Table S2. Both models were trained using Adam with learning rate 10^{-4} and weight decay 10^{-13} . Both models were trained on a single Tesla V100-PCIE-32GB GPU.

4. EVALUATION METHODOLOGY

A. Evaluation details

The main difference between our and other methods is that we generate 3D point clouds of atoms that are further connected with covalent bonds while other methods generate covalent bonds along with atom types. We emphasize that both DeLinker and 3DLinker employ valency rules at each generation step, which facilitates the generation of samples with high chemical validity. In our case, DiffLinker learns these chemical rules from the data and places atoms at the relevant distances from each other. Since the output of DiffLinker is a 3D point cloud, we compute covalent bonds between pairs of atoms based on their types and pairwise distances using OpenBabel [47].

To be consistent in the evaluation methodology, we recomputed covalent bonds using OpenBabel for all molecules in ZINC and CASF test sets. Next, for each updated molecule, we obtained linkers by removing irrelevant atoms and saved the resulting molecules and fragments in SDF and SMILES formats. Molecules saved in SDF format were considered as ground truth and used for 3D comparison (for computing RMSD and SC_{RDKit} metrics). Molecules and linkers saved in SMILES format were considered as ground truth and used for 2D comparison (novelty and recovery rates). To evaluate other methods, we used original SMILES representations.

Our samples

For each generated point cloud, we computed covalent bonds with OpenBabel, and extracted the largest connected component. Next, we obtained a linker by matching the generated molecule with the corresponding fragments (computed with OpenBabel as explained above) and removing irrelevant atoms. Finally, we kekulized [48] the resulting linker and saved the generated molecule with recomputed covalent bonds and the corresponding linker in SDF and SMILES formats.

Table S3. Average quantitative estimation of drug-likeness (QED) [49], synthetic accessibility (SA) [50], and number of rings for molecules in training, validation and test datasets.

	Dataset	QED \uparrow	SA \downarrow	# Rings \uparrow
ZINC	Train	0.734	2.986	0.213
	Validation	0.736	2.930	0.207
	Test	0.729	2.944	0.274
GEOM	Train	0.546	2.720	0.901
	Validation	0.556	2.711	0.853
	Test	0.537	2.735	0.849
Pockets	Train	0.330	5.330	0.926
	Validation	0.418	4.783	0.763
	Test	0.356	5.445	1.272
	CASF	0.481	3.572	0.271

Metrics

To compute validity, we apply sanitization and additionally check that the molecule contains all atoms from fragments. For all other metrics, we consider only a subset of valid samples. To compute novelty, we first preprocess SMILES of the linker by removing stereochemistry and using canonical tautomer SMILES. Next, we count how many of the resulting generated linker SMILES were represented in the training set. To compute uniqueness, we compare SMILES of whole molecules and compute the number of unique molecules sampled for each input pair of fragments. To compute recovery, we compare SMILES of each molecule sampled for a given pair of fragments with SMILES of the corresponding ground-truth molecule. Before the comparisons, we remove hydrogens and stereochemistry from molecules. To compute RMSD, we consider only recovered molecules and align them with the corresponding ground-truth molecules using RDKit function `rdkit.Chem.rdMolAlign` which returns the optimal RMSD for aligning two molecules. Other properties are also computed with the RDKit package, such as: quantitative drug-likeness (QED) [49]

using the function `rdkit.Chem.QED.qed`; synthetic accessibility (SA) score with the function `calculateScore` provided by Ertl and Schuffenhauer [50] in the RDKit-compatible package `sascorer.py`; number of rings, with the function `rdkit.Chem.rdMolDescriptors.CalcNumRings`. Table S3 provides mean QED, SA and number of rings computed for molecules in training, validation and test datasets.

B. 2D filters

2D Filters used by Imrie et al. [26] for constructing ZINC and CASF datasets include synthetic accessibility [50], ring aromaticity (RA), and pan-assay interference compounds (PAINS) [51] criteria. RA controls the correctness of the covalent bond orders in the rings of the a linker and PAINS checks if a molecule does not contain compounds that often give false-positive results in high-throughput screens [51]. Even though we used the same datasets as in [26] that were created using all three filters, we however modify the metric "Passed 2D Filters" by removing SA from it. Instead we introduce our own SA-based metric that we report separately.

Implementation of the SA filter

The molecule is considered to pass the synthetic accessibility filter if its SA-score [50] is lower than SA-score of the corresponding pair of fragments. Even though our models performed on par or better than DeLinker and 3DLinker according to all other metrics, almost all molecules generated by our models did not pass SA-filter. We investigated this issue and figured out that SMILES of fragments passed to SA filter by DeLinker and 3DLinker contained dummy atoms representing anchors. These atoms did not have any atom type assigned and therefore such molecules were considered hard to be synthesised. For almost all molecules in the test set SA-scores of fragments with dummy atoms were higher than SA-scores of the whole molecules. However, for most of fragments without dummy atoms SA-scores were much lower. Figure S1 shows 4 examples of molecules and fragments with and without dummy atoms and the corresponding SA-scores. We can conclude that SA-filter proposed by authors of DeLinker shows nothing but the fact that molecules with unknown atoms are hard to be synthesised. Therefore, we considered this metric to be irrelevant and excluded it from our report. Instead, we report average Synthetic Accessibility score of full generated molecules.

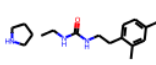



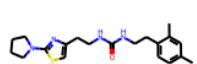
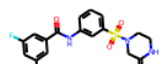
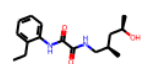
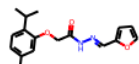
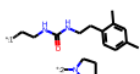
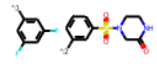


our fragments (without dummy atoms)	 2.11	 2.01	 2.11	 2.33
full molecules	 2.29	 2.23	 2.88	 2.21
DeLinker fragments (with dummy atoms)	 3.26	 3.49	 4.37	 3.44

Fig. S1. Synthetic accessibility scores (SA-scores) for fragments without dummy atoms (top row), full molecules (middle row) and fragments with dummy atoms (bottom row).

C. DeLinker and 3DLinker on GEOM Dataset

To evaluate DeLinker and 3DLinker on GEOM dataset, we had to filter the original test set consisting of 1,288 input fragment sets and remove examples with more than 3 disconnected fragments. For the remainder test set that included 1,170 input fragment triplets, we ran both DeLinker and 3DLinker twice: first, to connect two randomly selected fragments (10 samples per fragment pair) and then to connect the resulting compound with the third fragment (10 samples per input). To obtain the linker of the correct (ground-truth) size, in both steps, we generated half of the original linker size. Overall, we obtained 100 samples for each input fragment triplet. We used a pre-trained DeLinker model available at <https://github.com/oxpig/DeLinker> and a pre-trained 3DLinker model available at <https://github.com/YinanHuang/3DLinker>. The results are provided in Tables 1 (main text) and S5.

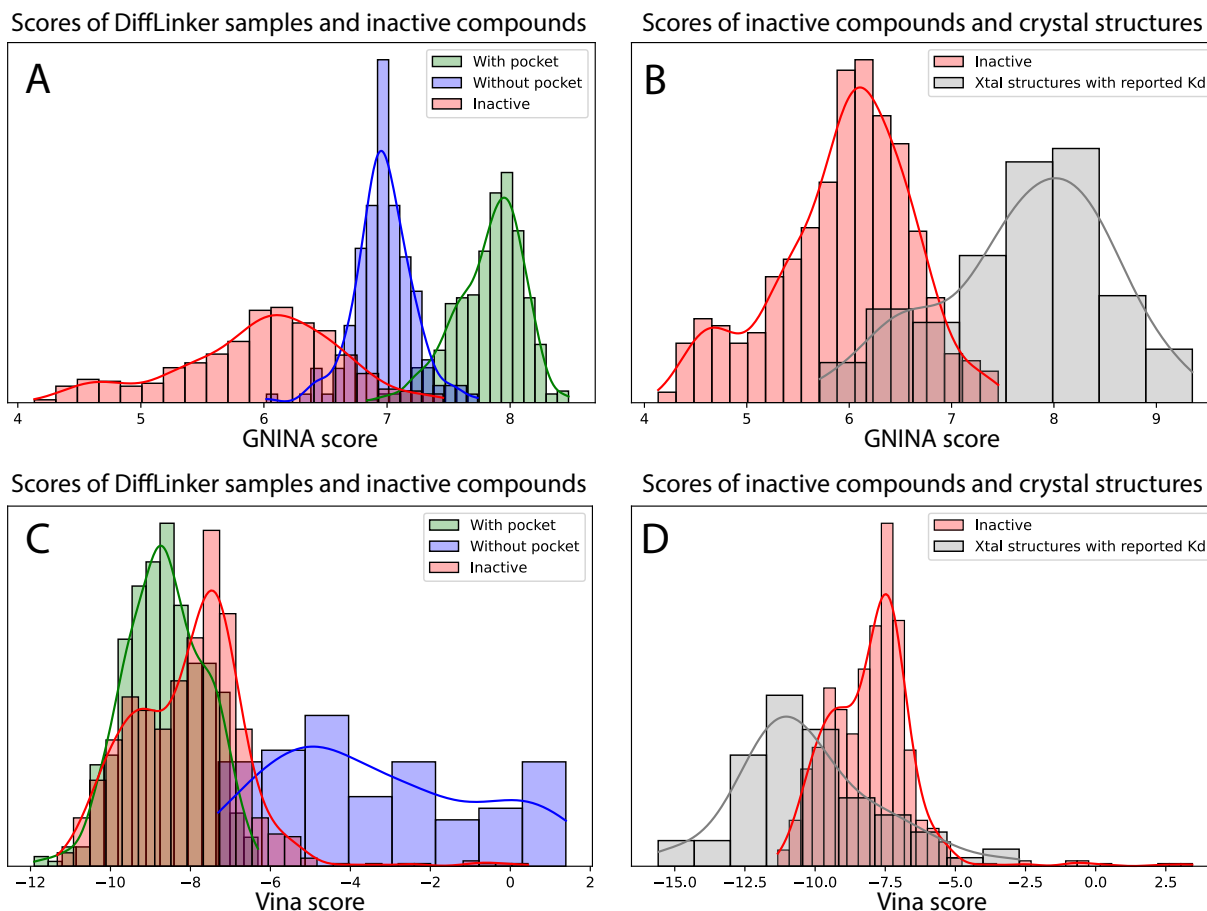


Fig. S2. Comparison of GNINA (the higher the better) and Vina (the lower the better) scores of inactive compounds with DiffLinker samples and crystal structures from PDBbind with reported K_D values.

D. GNINA for Hsp90 docking

To demonstrate the suitability of GNINA [52] and Vina for docking of small molecules to Hsp90 proteins, we perform two experiments.

First, we assemble a set of 76 Hsp90 structures with their corresponding ligands from PDBbind v2020 [53] database and compare the reported experimental K_D values to GNINA's and Vina predicted affinities without redocking or minimization in Extended Data Figure 3. While it is obvious that the computational docking tool is not a perfect predictor of *in vivo* binding affinity, there is some level of correlation between the experimental and computational values which justifies using GNINA as an accessible computational proxy of true binding affinity.

Second, we dock and score molecules reported to be inactive to Hsp90 in three binding assays (PubChem AID 754: 657 molecules, PubChem AID 687006: 81 molecules, PubChem AID 1803875: 18 molecules). Figure S2 shows the distributions of GNINA (A-B) and Vina (C-D) scores of the docked inactive compounds and compares them with the scores of DiffLinker samples (A-C) and ligands from PDBbind used in Figure 3G-H (main text). While both scores represent the expected situation where inactive compounds tend to have worse scores than other molecules, inactive molecules obtain relatively high Vina scores likely as a consequence of the docking being performed using the Vina scoring function.

While computational docking methods and state-of-the-art docking scores are known to have many limitations [54], experiments performed here with compounds inactive to Hsp90 (Figure S2) and with crystal structures of protein-ligand complexes with reported K_D values (Extended Data Figure 3) suggest that the chosen GNINA and Vina scores are satisfactory computational tools to assess the quality of samples generated by DiffLinker in terms of its interaction with the target protein.

5. ADDITIONAL RESULTS

A. GEOM

As mentioned in Section 4C, we filtered the original GEOM test set consisting of 1,288 input fragment sets and remove examples with more than 3 disconnected fragments. For each input fragment set, we obtained 100 samples with 3DLinker. For consistency we report DiffLinker performance on GEOM in Tables 1 (main text) and S5 computed in the same setting: 1,170 input examples and 100 samples per input. DiffLinker results computed on the full GEOM test set with 250 samples per input are provided in Tables S6 and S7.

B. Pockets

DiffLinker results on the Pockets test set with 100 samples per input are provided in Tables S6 and S7 as well. We note that train/test split of the Pockets dataset was performed solely based on PDB-codes of the protein-ligand complexes and EC numbers of proteins. In the resulting dataset used for evaluation, we therefore have 17 molecules that are also represented in the training set but bound to different proteins. For the full picture, we alternatively provide another reduced test set which does not contain molecules from the training set at all. It includes 453 examples from the initial test set. We provide evaluation metrics obtained on the reduced test set in Table S8.

C. Inpainting

Table S4. Comparison of DiffLinker trained in 3D-conditioning setting with DiffLinker trained in inpainting setting. Evaluation was performed on ZINC validation set. For each input pair of fragments we sampled 50 linkers.

Method	Valid, %	Recovered, %	RMSD ↓	SC _{RDKit} ↑	QED ↑	SA ↓	# Rings ↑
3D-conditioning	91.0	66.3	0.31	0.92	0.71	3.02	0.21
Inpainting	65.5	65.9	0.79	0.89	0.73	3.09	0.25

Instead of defining a 3D-conditional diffusion model, an alternative approach to this problem is to view linker generation as an inpainting task where a part of the molecule is known, and the rest needs to be recovered. Inpainting with diffusion models is achieved by training an unconditional diffusion model on the full molecules, and replacing the learned denoising process with the true denoising process for the known parts during sampling [46]. However, training a model on full molecules defeats the purpose of fragment-based molecule generation, which aims to reduce the problem size by avoiding the generation of all atoms. Instead of learning the full molecular space, which is necessary in inpainting formulation, we propose the 3D-conditioning mechanism that allows to focus on learning the subspace of valid linkers. Besides, inpainting formulation lacks consistency in case protein pockets are included: such a model would have to learn a more complex space of the molecules along with the corresponding pockets. At the same time, the proposed 3D-conditioning mechanism naturally scales to this scenario.

Technically, molecular linker design can be considered as an inpainting task, and such a strategy can be easily implemented using vanilla Equivariant Diffusion Model [31] with a minor adaptation of the sampling function. However, 3D-conditioning remarkably outperforms inpainting approach in the simplest linker design setup. In Table S4 we provide a comparison of DiffLinker trained in 3D-conditioning setting with DiffLinker trained in the inpainting setting. Both models have identical architectures and were trained with identical hyper parameters. For evaluation we used validation ZINC set (400 examples) and sampled 50 linkers for every pair of input fragments.

As shown in Table S4, using 3D-conditioned model reduces the number of invalid molecules (i.e., chemically incorrect or with disconnected fragments) by 25%. Moreover, the inpainting model generates molecules with worse RMSDs. Even though the chemical properties of the molecules generated by both methods are comparable, significantly lower validity indicates that inpainting approach is suboptimal in the implemented molecular linker design task.

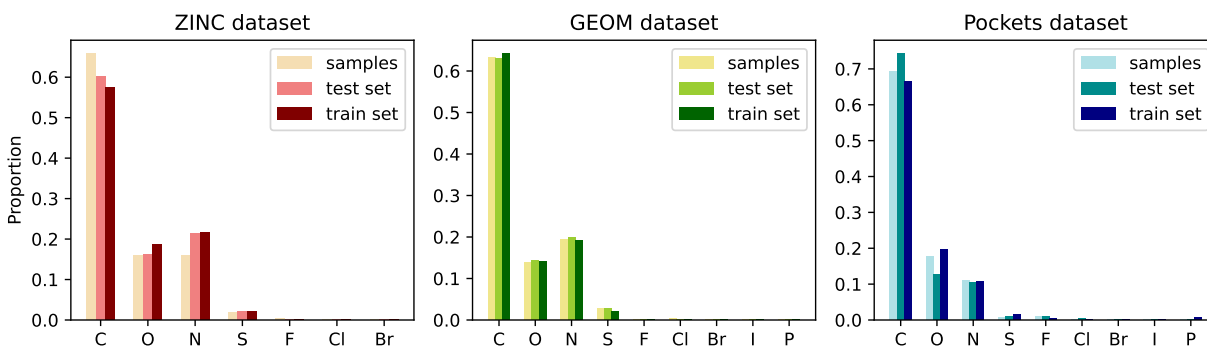


Fig. S3. Distributions of atom types in train and test datasets and the corresponding DiffLinker samples.

Table S5. Additional metrics assessing the ability to generate compounds that are similar to reference molecules. 2D filters include ring aromaticity (RA), and pan-assay interference compounds (PAINS) [51] criteria. SC_{RDKit} score reflects geometric and chemical similarity of samples to reference compounds. Top-2 best results for each metric are highlighted in bold.

Method	2D Filters, %	Recovery, %	RMSD ↓	SC_{RDKit}			Avg ↑	
				> 0.7	> 0.8	> 0.9		
ZINC	DeLinker + ConfVAE + MMFF	84.88	80.2	5.48	3.73	0.61	0.09	0.49
	3DLinker (given anchors)	84.24	94.0	0.10	99.86	97.05	63.78	0.92
	3DLinker	83.72	93.5	0.11	99.83	96.22	63.63	0.92
	DiffLinker	86.26	82.0	0.34	99.72	94.62	67.85	0.93
	DiffLinker (given anchors)	84.36	87.2	0.32	99.96	97.02	71.73	0.94
	DiffLinker (sampled size)	87.98	70.7	0.34	99.37	90.21	51.90	0.90
	DiffLinker (given anchors, sampled size)	84.76	77.5	0.35	99.67	95.04	56.35	0.91
CASF	DeLinker + ConfVAE + MMFF	77.25	52.8	11.89	1.24	0.19	0.03	0.39
	DiffLinker	87.73	42.8	0.44	92.17	79.62	50.14	0.84
	DiffLinker (given anchors)	82.37	50.2	0.37	95.07	89.05	60.63	0.86
	DiffLinker (sampled size)	89.27	40.5	0.34	92.83	79.12	43.99	0.82
	DiffLinker (given anchors, sampled size)	82.08	48.8	0.32	94.52	87.90	55.25	0.84
GEOM	DeLinker + ConfVAE + MMFF	100.00	0.0	—	0.00	0.00	0.00	0.42
	3DLinker	94.10	0.0	—	60.63	29.30	8.22	0.72
	DiffLinker	47.68	85.6	0.12	95.53	87.87	71.36	0.92
	DiffLinker (given anchors)	49.70	85.2	0.12	95.12	86.79	69.91	0.92
	DiffLinker (sampled size)	55.68	68.8	0.07	91.74	81.47	57.11	0.88
	DiffLinker (given anchors, sampled size)	57.90	67.9	0.07	90.95	79.57	54.78	0.87

Table S6. Performance metrics on GEOM (250 samples, full test set) and Pockets test set (100 samples). The first three metrics, average quantitative estimation of drug-likeness (QED) [49], average synthetic accessibility (SA) [50] and average number of rings in the linker, assess the chemical relevance of the generated molecules. The last three metrics, validity, uniqueness and novelty, evaluate the standard generative properties of the methods. Top-2 best results for each metric are highlighted in bold.

	Method	QED \uparrow	SA \downarrow	# Rings \uparrow	Valid, %	Unique, %	Novel, %
GEOM	DiffLinker	0.48	2.99	0.75	93.4	31.6	68.7
	DiffLinker (given anchors)	0.49	3.01	0.79	93.5	32.1	68.5
	DiffLinker (sampled size)	0.45	3.27	0.76	87.1	57.3	76.2
	DiffLinker (given anchors, sampled size)	0.46	3.33	0.84	88.6	58.2	76.2
Pockets	DiffLinker (pocket atoms)	0.45	3.89	1.06	88.7	62.5	73.1
	DiffLinker (pocket backbone)	0.45	3.77	0.95	90.4	60.9	72.8
	DiffLinker (unconditioned)	0.45	3.83	1.10	93.6	61.1	74.9

Table S7. Additional metrics assessing the ability to generate compounds that are similar to reference molecules. For GEOM (250 samples, full test set) and Pockets test set (100 samples). 2D filters include ring aromaticity (RA), and pan-assay interference compounds (PAINS) [51] criteria. SC_{RDKit} score reflects geometric and chemical similarity of samples to reference compounds. Top-2 best results for each metric are highlighted in bold.

	Method	2D Filters, %	Recovery, %	RMSD \downarrow	SC_{RDKit}			Avg \uparrow
					> 0.7	> 0.8	> 0.9	
GEOM	DiffLinker	49.47	89.1	0.11	95.90	88.81	73.31	0.93
	DiffLinker (given anchors)	51.25	88.0	0.11	95.57	87.77	72.12	0.93
	DiffLinker (sampled size)	56.05	77.5	0.07	92.11	82.11	58.03	0.88
	DiffLinker (given anchors, sampled size)	58.16	77.1	0.07	91.38	80.37	55.71	0.88
Pockets	DiffLinker (pocket atoms)	69.47	37.3	0.87	71.61	57.30	34.52	0.78
	DiffLinker (pocket backbone)	66.48	37.8	0.89	65.58	52.77	31.50	0.76
	DiffLinker (unconditioned)	59.43	36.5	0.89	64.06	50.95	30.59	0.75

Table S8. Performance metrics on the reduced Pockets test set (453 examples, 100 samples). Average quantitative estimation of drug-likeness (QED) [49], average synthetic accessibility (SA) [50] and average number of rings in the linker assess the chemical relevance of the generated molecules. Validity, uniqueness and novelty, evaluate the standard generative properties of the methods. 2D filters include ring aromaticity (RA), and pan-assay interference compounds (PAINS) [51] criteria. SC_{RDKit} score reflects geometric and chemical similarity of the samples to the reference compounds. Top-2 best results for each metric are highlighted in bold.

Method	QED \uparrow	SA \downarrow	# Rings \uparrow	Valid, %	Unique, %	Novel, %
DiffLinker (pocket atoms)	0.50	3.77	1.07	86.2	77.5	90.7
DiffLinker (pocket backbone)	0.50	3.62	0.94	88.3	75.6	90.6
DiffLinker (unconditioned)	0.51	3.70	1.13	92.0	75.7	93.0

Method	2D Filters, %	Recovery, %	RMSD \downarrow	SC_{RDKit}			
				> 0.7	> 0.8	> 0.9	Avg \uparrow
DiffLinker (pocket atoms)	60.84	31.2	0.86	66.35	48.67	26.62	0.75
DiffLinker (pocket backbone)	57.15	31.1	1.03	58.76	42.81	23.14	0.72
DiffLinker (unconditioned)	48.52	30.1	0.96	57.03	40.57	22.16	0.72

Table S9. Standard deviation of QED, SA, number of rings, RMSD and SC_{RDKit} metrics reported in Tables 1 (main text), S5, S6 and S7

	Dataset	QED	SA	# Rings	RMSD	SC_{RDKit}
ZINC	DeLinker + ConfVAE + MMFF	0.16	0.68	0.42	2.29	0.14
	3DLinker (given anchors)	0.16	0.67	0.42	0.29	0.08
	3DLinker	0.16	0.68	0.43	0.31	0.08
	DiffLinker	0.14	0.73	0.45	0.44	0.07
	DiffLinker (given anchors)	0.14	0.76	0.47	0.43	0.06
	DiffLinker (sampled size)	0.15	0.77	0.54	0.44	0.08
	DiffLinker (given anchors, sampled size)	0.15	0.81	0.59	0.49	0.08
CASF	DeLinker + ConfVAE + MMFF	0.18	0.95	0.66	5.30	0.13
	DiffLinker	0.18	0.90	0.60	0.53	0.28
	DiffLinker (given anchors)	0.18	0.90	0.66	0.42	0.27
	DiffLinker (sampled size)	0.16	0.87	0.53	0.43	0.29
	DiffLinker (given anchors, sampled size)	0.17	0.88	0.61	0.41	0.30
GEOM	DeLinker + ConfVAE + MMFF	0.13	0.88	0.00	—	0.07
	3DLinker	0.16	0.61	0.00	—	0.14
	DiffLinker	0.14	0.62	0.73	0.29	0.12
	DiffLinker (given anchors)	0.14	0.64	0.78	0.27	0.12
	DiffLinker (sampled size)	0.15	0.70	0.87	0.18	0.16
	DiffLinker (given anchors, sampled size)	0.14	0.74	0.95	0.20	0.15
Pockets	DiffLinker (pocket atoms)	0.20	0.70	0.89	0.85	0.21
	DiffLinker (pocket backbone)	0.19	0.68	0.75	0.86	0.21
	DiffLinker (unconditioned)	0.20	0.68	0.85	0.82	0.22

Table S10. Sampling time for different datasets (with $T = 500$ denoising steps). Experiments were performed on a single Tesla V100-PCIE-32GB GPU.

dataset	batch size	time per molecule, s
ZINC	128	0.37
CASF	128	0.63
Multi-Fragment	64	0.37
Pockets (full atomic representation)	32	0.46
Pockets (backbone representation)	32	0.21
Pockets (no pocket)	32	0.12

REFERENCES

1. A. Bancet, C. Raingeval, T. Lomberget, *et al.*, "Fragment linking strategies for structure-based drug design," *J. Medicinal Chem.* **63**, 11420–11435 (2020).
2. C. De Fusco, P. Brear, J. Iegre, *et al.*, "A fragment-based approach leading to the discovery of a novel binding site and the selective ck2 inhibitor cam4066," *Bioorganic & medicinal chemistry* **25**, 3471–3482 (2017).
3. A. Kohlmann, S. G. Zech, F. Li, *et al.*, "Fragment growing and linking lead to novel nanomolar lactate dehydrogenase inhibitors," *J. medicinal chemistry* **56**, 1023–1040 (2013).
4. H. Mobitz, R. Machauer, P. Holzer, *et al.*, "Discovery of potent, selective, and structurally novel dot11 inhibitors by a fragment linking approach," *ACS medicinal chemistry letters* **8**, 338–343 (2017).
5. M. Békés, D. R. Langley, and C. M. Crews, "Protac targeted protein degraders: the past is prologue," *Nat. Rev. Drug Discov.* **21**, 181–200 (2022).
6. N. Bai, S. A. Miller, G. V. Andrianov, *et al.*, "Rationalizing protac-mediated ternary complex formation using rosetta," *J. chemical information modeling* **61**, 1368–1382 (2021).
7. W. Farnaby, M. Koegl, M. J. Roy, *et al.*, "Baf complex vulnerabilities in cancer demonstrated via structure-based protac design," *Nat. chemical biology* **15**, 672–680 (2019).
8. H. Sun, G. Tawa, and A. Wallqvist, "Classification of scaffold-hopping approaches," *Drug discovery today* **17**, 310–324 (2012).
9. C. Sheng and W. Zhang, "Fragment informatics and computational fragment-based drug design: an overview and update," *Medicinal Res. Rev.* **33**, 554–598 (2013).
10. V. Tschinke and N. C. Cohen, "The newlead program: a new method for the design of candidate structures from pharmacophoric hypotheses," *J. medicinal chemistry* **36**, 3863–3870 (1993).
11. A. Miranker and M. Karplus, "An automated method for dynamic ligand design," *Proteins: Struct. Funct. Bioinforma.* **23**, 472–490 (1995).
12. D. C. Roe and I. D. Kuntz, "Builder v. 2: improving the chemistry of a de novo design strategy," *J. Comput. Mol. Des.* **9**, 269–282 (1995).
13. D. A. Pearlman and M. A. Murcko, "Concerts: dynamic connection of fragments as an approach to de novo ligand design," *J. medicinal chemistry* **39**, 1651–1663 (1996).
14. M. Stahl, N. P. Todorov, T. James, *et al.*, "A validation study on the practical use of automated de novo design," *J. computer-aided molecular design* **16**, 459–478 (2002).
15. D. C. Thompson, R. Aldrin Denny, R. Nilakantan, *et al.*, "Confirm: connecting fragments found in receptor molecules," *J. Comput. Mol. Des.* **22**, 761–772 (2008).
16. O. Ichihara, J. Barker, R. J. Law, and M. Whittaker, "Compound design by fragment-linking," *Mol. Informatics* **30**, 298–306 (2011).
17. H.-J. Böhm, "The computer program ludi: a new method for the de novo design of enzyme inhibitors," *J. computer-aided molecular design* **6**, 61–78 (1992).
18. G. Lauri and P. A. Bartlett, "Caveat: a program to facilitate the design of organic molecules," *J. computer-aided molecular design* **8**, 51–66 (1994).
19. F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. & operations research* **13**, 533–549 (1986).
20. F. Dey and A. Caflisch, "Fragment-based de novo ligand design by multiobjective evolutionary optimization," *J. chemical information modeling* **48**, 679–690 (2008).
21. H. Ji, W. Zhang, M. Zhang, *et al.*, "Structure-based de novo design, synthesis, and biological evaluation of non-azole inhibitors specific for lanosterol 14 α -demethylase of fungi," *J. medicinal chemistry* **46**, 474–485 (2003).
22. R. B. Silverman, "Design of selective neuronal nitric oxide synthase inhibitors for the prevention and treatment of neurodegenerative diseases," *Accounts chemical research* **42**, 439–451 (2009).
23. C. Sheng and W. Zhang, "New lead structures in antifungal drug discovery," *Curr. medicinal chemistry* **18**, 733–766 (2011).
24. Y. Yang, S. Zheng, S. Su, *et al.*, "Syntalinker: automatic fragment linking with deep conditional transformer neural networks," *Chem. science* **11**, 8312–8322 (2020).
25. G. Zweig, J. C. Platt, C. Meek, *et al.*, "Computational approaches to sentence completion," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Association for Computational Linguistics, Jeju Island, Korea, 2012), pp. 601–610.
26. F. Imrie, A. R. Bradley, M. van der Schaar, and C. M. Deane, "Deep generative models for 3d linker design," *J. chemical information modeling* **60**, 1983–1995 (2020).
27. F. Imrie, T. E. Hadfield, A. R. Bradley, and C. M. Deane, "Deep generative design with 3d pharmacophoric constraints," *Chem. science* **12**, 14577–14589 (2021).
28. Y. Huang, X. Peng, J. Ma, and M. Zhang, "3dlinker: An e(3) equivariant variational autoencoder for molecular linker design," *arXiv preprint arXiv:2205.07309* (2022).
29. B. Elesedy and S. Zaidi, "Provably strict generalisation benefit for equivariant models," in *International Conference on*

- Machine Learning*, (PMLR, 2021), pp. 2959–2969.
30. M. Rath and A. P. Condurache, “Improving the sample-complexity of deep classification networks with invariant integration,” arXiv preprint arXiv:2202.03967 (2022).
 31. E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling, “Equivariant diffusion for molecule generation in 3d,” in *International Conference on Machine Learning*, (PMLR, 2022), pp. 8867–8887.
 32. C. Vignac, I. Krawczuk, A. Siraudin, *et al.*, “Digress: Discrete denoising diffusion for graph generation,” arXiv preprint arXiv:2209.14734 (2022).
 33. M. Xu, L. Yu, Y. Song, *et al.*, “Geodiff: A geometric diffusion model for molecular conformation generation,” arXiv preprint arXiv:2203.02923 (2022).
 34. C. Shi, S. Luo, M. Xu, and J. Tang, “Learning gradient fields for molecular conformation generation,” in *International Conference on Machine Learning*, (PMLR, 2021), pp. 9558–9568.
 35. B. Jing, G. Corso, J. Chang, *et al.*, “Torsional diffusion for molecular conformer generation,” arXiv preprint arXiv:2206.01729 (2022).
 36. G. Corso, H. Stärk, B. Jing, *et al.*, “Diffdock: Diffusion steps, twists, and turns for molecular docking,” arXiv preprint arXiv:2210.01776 (2022).
 37. A. Schneuing, Y. Du, C. Harris, *et al.*, “Structure-based drug design with equivariant diffusion models,” arXiv preprint arXiv:2210.13695 (2022).
 38. H. Lin, Y. Huang, M. Liu, *et al.*, “Diffbp: Generative diffusion of 3d molecules for target protein binding,” arXiv preprint arXiv:2211.11214 (2022).
 39. S. Lu, L. Yao, X. Chen, *et al.*, “3d molecular generation by virtual dynamics,” (2023).
 40. B. L. Trippe, J. Yim, D. Tischer, *et al.*, “Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem,” arXiv preprint arXiv:2206.04119 (2022).
 41. S. Luo, Y. Su, X. Peng, *et al.*, “Antigen-specific antibody design and optimization with diffusion-based generative models,” bioRxiv (2022).
 42. Z. C. V. F. A. I. S. T. W. W. V. X. F. O. A. B. G. G. John Ingraham, Max Baranov, “Illuminating protein space with a programmable generative model,” (2022).
 43. J. L. Watson, D. Juergens, N. R. Bennett, *et al.*, “Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models,” bioRxiv pp. 2022–12 (2022).
 44. Z. Qiao, W. Nie, A. Vahdat, *et al.*, “Dynamic-backbone protein-ligand structure prediction with multiscale generative diffusion models,” arXiv preprint arXiv:2209.15171 (2022).
 45. J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” in *International Conference on Machine Learning*, (PMLR, 2015), pp. 2256–2265.
 46. A. Lugmayr, M. Danelljan, A. Romero, *et al.*, “Repaint: Inpainting using denoising diffusion probabilistic models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2022), pp. 11461–11471.
 47. N. M. O’Boyle, M. Banck, C. A. James, *et al.*, “Open babel: An open chemical toolbox,” *J. cheminformatics* **3**, 1–14 (2011).
 48. A. Kekulé, “Sur la constitution des substances aromatiques,” *Bull. mensuel de la Société Chimique de Paris* **3**, 98 (1865).
 49. G. R. Bickerton, G. V. Paolini, J. Besnard, *et al.*, “Quantifying the chemical beauty of drugs,” *Nat. chemistry* **4**, 90–98 (2012).
 50. P. Ertl and A. Schuffenhauer, “Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions,” *J. cheminformatics* **1**, 1–11 (2009).
 51. J. B. Baell and G. A. Holloway, “New substructure filters for removal of pan assay interference compounds (pains) from screening libraries and for their exclusion in bioassays,” *J. medicinal chemistry* **53**, 2719–2740 (2010).
 52. A. T. McNutt, P. Francoeur, R. Aggarwal, *et al.*, “Gnina 1.0: molecular docking with deep learning,” *J. cheminformatics* **13**, 1–20 (2021).
 53. R. Wang, X. Fang, Y. Lu, *et al.*, “The pdbbind database: methodologies and updates,” *J. medicinal chemistry* **48**, 4111–4119 (2005).
 54. C. Harris, K. Didi, A. R. Jamasb, *et al.*, “Benchmarking generated poses: How rational is structure-based drug design with generative models?” arXiv preprint arXiv:2308.07413 (2023).