Improving Generative Models for 3D Molecular Structures

by

Ameya Daigavane

B.Tech., Indian Institute of Technology Guwahati (2020)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by:	Ameya Daigavane Department of Electrical Engineering and Computer Science May 17, 2024
Certified by:	Tess E. Smidt Assistant Professor of Electrical Engineering and Computer Science Thesis Supervisor
Accepted by:	Leslie A. Kolodziejski Professor of Electrical Engineering and Computer Science Chair, Department Committee on Graduate Students

Improving Generative Models for 3D Molecular Structures

Ameya Daigavane

ABSTRACT

Generative models have recently emerged as a promising avenue for navigating the high-dimensional space of molecular structures. Such models must be designed carefully to respect the rotation and translation symmetries of molecules. In this thesis, we first provide an overview of existing methods and techniques in this rapidly developing field. Next, we present Symphony, an $E(3)$-equivariant autoregressive generative model for 3D molecular geometries that iteratively builds a molecule from molecular fragments, improving upon existing autoregressive models for molecule generation and approaching the performance of diffusion models.

The material in this thesis is primarily sourced from the publication “Symphony: Symmetry-Equivariant Point-Centered Spherical Harmonics for 3D Molecule Generation" [13] authored by Ameya Daigavane, Song Kim, Mario Geiger and Tess Smidt, and published at the International Conference on Learning Representations (ICLR), 2024.

Acknowledgments

The work in this SM thesis represents a milestone in a journey that has only just begun. I came to MIT in 2022, knowing almost nothing about $E(3)$-equivariance but excited to work on scientific problems again. Almost two years later, I have learned so much and I understand some of these things a little better. The next few years as a PhD student are going to be even more promising!

I would like to thank my advisor Prof. Tess Smidt and the wonderful Atomic Architects group of which I am a proud member of. I have learned so much from all of you. To YuQing, Elyssa, Adriana, Jin, Allan, Ilan, Hannah, Mit, Song, Tuong, Yi-Lun, Julia, Max, Giulianna, Aria, Ray and Mario, thank you for sharing a little bit of your life with me in the lab. To Arijit, Prerna, Harsha, Mehul, Anushka, Jerry and Liad, thank you for keeping me sane outside of lab. Finally, to my family, Vaishali, Shrikant, Naveen, Ramyani and Minoli, thank you for the constant love, support, encouragement and patience that you have given me. This thesis would not have been possible without all of you.

Most of this thesis stems from my first research project at MIT with Song Kim, Mario Geiger and my advisor, Tess Smidt. Their contributions to this manuscript are invaluable, and I am incredibly grateful for the opportunity to be here, at this time, to have worked with them.

I would like to thank the National Science Foundation (NSF) Graduate Research Fellowship (GRFP) program and the NSF AI Institute for Artificial Intelligence and Fundamental Interactions (IAIFI) for their funding.

Contents

Abstract 2

Acknowledgments 3

List of Figures 7

List of Tables 9

1 Introduction 11

2 Related Work 12
- 2.1 Diffusion and Flow-Matching Models . . . 12
- 2.2 Autoregressive Models . . . 14

3 Background 18
- 3.1 $E(3)$-Equivariant Features . . . 18
- 3.2 Spherical Harmonics . . . 18

4 Methods 20
- 4.1 Building Molecules via Sequences of Fragments . . . 20
- 4.2 Handling the Symmetries of Fragments . . . 21
- 4.3 The Design of Symphony . . . 22
- 4.4 Bypassing the Angular Frequency Bottleneck . . . 23
- 4.5 Training and Inference . . . 23

5 Experimental Results 25
- 5.1 Validity of Generated Structures . . . 25
- 5.2 Capturing Training Set Statistics . . . 26
- 5.3 Generalization Capabilities . . . 27
- 5.4 Molecule Generation Throughput . . . 27
- 5.5 Statistics of Generated Molecules . . . 28
  - 5.5.1 Bispectra of Local Environments in Sampled Molecules . . . 28
  - 5.5.2 Bond Lengths in Sampled Molecules . . . 29
  - 5.5.3 Atom Type Counts . . . 29
  - 5.5.4 Ring Sizes . . . 29
- 5.6 Visualizing Generated Molecules . . . 34

6 Conclusion **35**

A Proof of Equivariance **36**
A.1 $E(3)$-Equivariance . . . 36
A.2 Permutation-Equivariance . . . 37

B The Advantage of Using Multiple Channels of Spherical Harmonics **38**
B.1 An Example with the Octahedron . . . 38
B.2 A Study on Learning Random Signals . . . 39

C Learning and Sampling from Position Distributions **41**
C.1 Learning Spherical Harmonic Coefficients . . . 41
C.2 Sampling from the Learned Position Distribution . . . 42
C.3 Representing Dirac Delta Distributions . . . 43

D Ablation Studies **44**
D.1 Ablation: l_{max} and Number of Position Channels . . . 44
D.2 Ablation: Training and Sampling Resolution . . . 45
D.3 Ablation: Sampling Temperature . . . 46

E Details for Reproducibility **48**
E.1 Details of Models . . . 48
E.1.1 Embedders . . . 48
E.1.2 Training Details . . . 49
E.1.3 Data Details . . . 49
E.1.4 Baseline Model Details . . . 50
E.2 Details of Metrics . . . 51
E.2.1 PoseBusters . . . 51
E.2.2 Maximum Mean Discrepancy . . . 51

List of Figures

2.1 Diffusion and flow-matching models work by gradually denoising a noisy version of a 3D molecular structure. . . . 12
2.2 Atom stability on QM9 as measured by Hoogeboom et al. [20] as a function of radial cutoff c in angstroms, showing a sharp drop in performance as the cutoff is decreased. . . . 14
2.3 Autoregressive models gradually build up a molecule atom-by-atom. . . . 14
2.4 $E(3)$-equivariance under rotations $\mathbf{R}$ and translations $\mathbf{T}$ for each marginal distribution $p(x_i|x_0, \ldots, x_{i-1})$ guarantees the $E(3)$-equivariance of $p(\{x_i\}_{i=1}^n)$. . . . 15
2.5 Defining the target position relative to a focus atom guarantees translational invariance. . . . 15
2.6 The triangulation procedure to place the next atom as depicted in Simm et al. [47] and G-SphereNet [30]. . . . 16

3.1 Plots of $Y_{l,m}(\theta, \phi)$ as l varies from 0 to 2, and m varies from $-l$ to l. The radial component and color intensity is proportional to the amplitude of $Y_{l,m}(\theta, \phi)$ at each point $(1, \theta, \phi)$. . . . 19
3.2 Decomposing a function into a linear combination of spherical harmonics. . . . 19

4.1 Fragments from CreateFragmentSequence applied to methane (CH4). Note the two options available to complete the initial fragment $\mathcal{S}^n$. . . . 21
4.2 One iteration of the Symphony molecular generation process, in which one atom is sampled given the positions and atom types of an unfinished molecular fragment $\mathcal{S}^n$ to create the next fragment $\mathcal{S}^{n+1}$. . . . 22

5.1 Bispectra of local environments of type C: C2,H2 and type C: C1,H3 respectively. Each row corresponds to a sample of the bispectrum (an array of length 15). Every entry of the bispectra is colored by value according to the colorbar on the right. . . . 28
5.2 Histogram of bond lengths for the five most frequent bonds in QM9. . . . 30
5.3 Histogram of bond lengths for the sixth to tenth most frequent bonds in QM9. . . . 31
5.4 Frequency of atom type counts in generated molecules on a log-scale. . . . 32
5.5 Frequency of ring sizes in generated molecules on a log-scale. . . . 32
5.6 Molecules generated by Symphony and visualized with PyMOL [43]. . . . 34

B.1 Usually, we would require $l_{\max} = 4$ to represent p^{pos} for the 'stars' and 'square' atoms, centered at the red central atom. With two channels, we only need up to $l_{\max} = 2$ each. . . . 38

B.2 Final KL divergence $KL(q \mid\mid p_c)$ for learned coefficients c as a function of number of position channels ch and $l_{\max}$. 40
B.3 Final KL divergence $KL(q \mid\mid p_c)$ for learned coefficients c as a function of number of position channels ch and $l_{\max}$, with the parametrization proposed by Simm et al. [49]. Removing the regularization term helps the model learn better. 40

D.1 Validity as a function of $l_{\max}$ for the position and focus embedders. Models for which $l_{\max} = 1$ for the focus embedder are marked in blue. Models for which $l_{\max} = 2$ for the focus embedder are marked in red. The intensity of colours increases with the number of position channels. 44
D.2 Validity as a function of sampling grid resolution (r_θ, r_ϕ). 45
D.3 The effect of resolution when learning the random signal from section B.2. Our original model was trained with a resolution of $(r_\theta, r_\phi) = (180, 359)$. 46
D.4 Validity as a function of temperature applied to the focus (above) and position (below) distribution logits. 47

List of Tables

5.1 Validity and uniqueness (among valid) percentages of molecules with different bond assignment methods, with **best** and second-best models highlighted. 26

5.2 Percentage of valid (as obtained from `xyz2mol`) molecules passing each PoseBusters test. 26

5.3 Comparing statistics of generated molecules to those found in QM9. (Top): The MMD of bond lengths for the 10 most frequent bonds. The notation 'X-Y: T' means that a X atom was bonded to a Y atom with a bond of type T. (Middle): The MMD of bispectra for the 10 most occurring local environments. The notation 'X: Y*n*,Z*m*' means that an X atom was the central atom, surrounded by *n* Y atoms and *m* Z atoms. (Bottom): The JSD of occurrence counts for atom types and local environments. ↓ indicates that lower is better for the metrics. 33

5.4 Comparing the difference between fragment completion rates on (seen) training and (unseen) testing fragments with one hydrogen removed. 33

E.1 Description of each intramolecular PoseBusters test, taken from Table 4 of Buttenschoen et al. [10]. 51

Chapter 1

Introduction

In silico generation of atomic systems with diverse geometries and desirable properties is important to many areas including fundamental science, materials design, and drug discovery [4]. The direct enumeration and validation of all possible 3D structures is computationally infeasible and does not in itself lead to useful representations of atomic systems for guiding understanding or design. Thus, there is interest in 'generative models' that can generate 3D molecular structures using machine learning algorithms.

Effective generative models of atomic systems must learn to represent and produce highly-correlated geometries that represent chemically valid and energetically favorable configurations. To do this, they must overcome several challenges:

1. The validity of an atomic system is ultimately determined by quantum mechanics. Generative models of atomic systems are trained on 3D structures relaxed through computationally-intensive quantum mechanical calculations. These models must learn to adhere to chemical rules, generating stable molecular structures based solely on examples.

2. The stability of atomic systems hinges on the precise placement of individual atoms. The omission or misplacement of a single atom can result in significant property changes and instability.

3. Atomic systems have inherent symmetries. Atoms of the same element are indistinguishable, so there is no consistent way to order atoms within an atomic system. Additionally, atomic systems lack unique coordinate systems (global symmetry) and recurring geometric patterns occur in a variety of locations and orientations (local symmetry).

Taking these challenges into consideration, the majority of generative models for atomic systems operate on point geometries and use permutation and Euclidean symmetry-invariant or equivariant methods. Thus far, two approaches have been emerged as effective for directly generating general 3D geometries of molecular systems: autoregressive models [15, 16, 30, 47, 49] and end-to-end diffusion and flow-matching models [20, 31, 51].

A related (but simpler) problem is that of molecular conformer generation, where a 3D conformation must be generated corresponding to a 2D molecular graph. Many approaches for generating such conformers based on the probabilistic modelling techniques discussed here have emerged over the past few years [14, 19, 22, 46, 48, 53, 55, 57–59], but the problem setting remains quite different.

Chapter 2

Related Work

Here, we provide an overview of relevant techniques for 3D molecular generation.

2.1 Diffusion and Flow-Matching Models

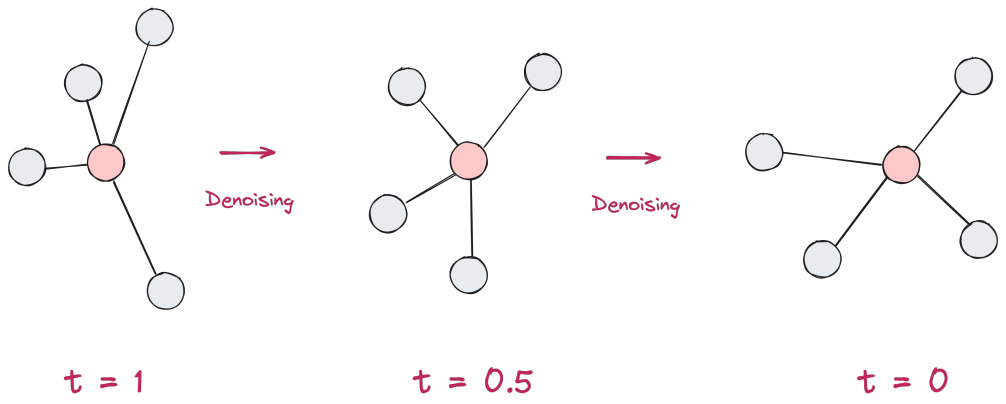

Figure 2.1: Diffusion and flow-matching models work by gradually denoising a noisy version of a 3D molecular structure.

Diffusion models describe the 3D structure of a molecule as a matrix $x_0 \in \mathbb{R}^{N(d+3)}$, where we specify the 3 spatial coordinates and d auxiliary features (indicating atom types and charges, for example) for each of the N atoms. In practice, N is sampled from a learned distribution based on the training set. Of course, the ordering of the N atoms is arbitrary, and generative models should ideally be invariant to permutations of these N atoms. Further, because the molecule lives in 3D space, the action of rotations and translations on this group must also be respected. As defined in Hoogeboom et al. [20], this is termed $E(3)$-equivariance of the learned probability distribution over 3D structures M:

$$p_{\text{model}}(gM) = \rho(g)p_{\text{model}}(M) \quad \text{for all} \quad g \in E(3), M \in \mathcal{M} \tag{2.1}$$

As discussed, in the context of diffusion models, $\mathcal{M} = \mathbb{R}^{N(d+3)}$.

$E(3)$ refers to the Euclidean group in 3 dimensions, consisting of all rotations, translations and reflections of 3D space. (In case the enantiomers of a chiral molecule need to be treated differently, our models should instead be $SE(3)$-equivariant, where reflections do not play a role.)

Diffusion models can be described as trying to learn the time-reversal of a stochastic process.

Let $x(t) \in \mathbb{R}^{N(d+3)}$ denote the 'noisy' state at time t. With $x(0) = x_0$, the state follows the stochastic differential equation (SDE) given by the Itô process:

$$dx = f(x,t)dt + G(t)dw$$

where $f : \mathbb{R}^{N(d+3)} \times \mathbb{R}^+ \to \mathbb{R}^{N(d+3)}$ and $G : \mathbb{R}^+ \to \mathbb{R}^+$ are the known 'drift' and 'diffusion' functions. w is a standard vector Weiner process with $w_t \in \mathbb{R}^{N(d+3)}$. The corresponding reverse-time SDE was derived by Anderson [3]:

$$d\bar{x} = (f(x,t) - G(t)^2 \nabla_x \log p_t(x))dt + G(x)d\bar{w}$$

As demonstrated by Song et al. [50], one can construct f and G such that as time $t \to \infty$, the distribution $p_t(x)$ converges to a fixed prior distribution.

Diffusion models aim to learn the score function $s_\theta(x,t) \approx \nabla_x \log p_t(x)$, which allows simulating the process in reverse, starting from samples from the prior distribution. In practice, this SDE is solved numerically (for example, with the Euler–Maruyama scheme), starting from some large but finite time T.

To ensure the $E(3)$-equivariance property (Equation 2.1), it is sufficient to ensure that the score function is $E(3)$-equivariant and the prior distribution is rotationally-invariant. Translations are handled by representing all positions in the center-of-mass (CoM) frame. $E(3)$-equivariant neural networks have become a popular choice for modelling the score function; many network designs for guaranteeing $E(3)$-equivariance with graph neural networks (GNNs) [12, 40] have been designed recently [2, 6–8, 17, 33, 41, 45].

The first successful application of diffusion models for the task of generating 3D molecular structures was performed by Hoogeboom et al. [20], with several improvements to the noise scheduler and geometric message-passing schemes followed up by Le et al. [28], Morehead and Cheng [31], Vignac et al. [54].

Flow-matching [29] has emerged as an alternative approach to diffusion. In flow-matching, one attempts to learn the velocity field of an ordinary differential equation (ODE) transforming noise to data, which is significantly easier to sample from [1]. The application of flow-matching to molecular generation was first demonstrated by Song et al. [51], showing competitive results with diffusion models.

The main limitation of both diffusion and flow-matching models is the requirement of using fully-connected graphs in the underlying GNN model for the score prediction. Furthermore, diffusion models are significantly slower to sample from, because the underlying neural network is invoked ≈ 1000 times when sampling a single molecule. This limits the size of the systems that can be represented by these models. An attractive option is to apply a distance-based cutoff c to the edges:

$$(i,j) \in E \iff ||\vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j|| \leq c.$$

In Figure 2.2, we measured the 'atom stability' as computed by Hoogeboom et al. [20] on the EDM-generated molecules as the cutoff c is varied when trained on QM9. The legend shows the

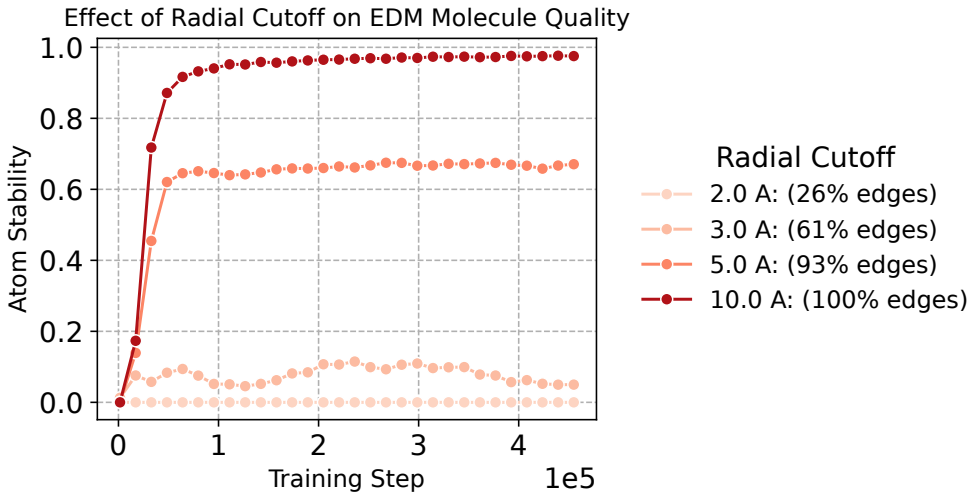

Figure 2.2: Atom stability on QM9 as measured by Hoogeboom et al. [20] as a function of radial cutoff c in angstroms, showing a sharp drop in performance as the cutoff is decreased.

percentage of edges in the original graph that are preserved by the cutoff. We notice a sharp drop-off in this metric as the cutoff c (and hence, the number of edges) reduces, despite the use of many message-passing layers in the underlying graph neural network. Even for cutoff $c = 5$ A, where 93% of the edges in the graph still remain, the quality of the EDM-generated molecules drops significantly. On observing the trajectories of the individual atoms, we noticed that the system tends to explode for lower values of cutoff c. Thus, scaling up diffusion and flow-matching models to large systems remains an important open problem. Randomized graph construction, such as long-range graph connections explored by Chroma [21] for protein design, could be a path forward. Distillation of diffusion models as introduced by Salimans and Ho [39] to reduce the number of sampling steps has been explored in the molecular generation space as well [23, 26].

2.2 Autoregressive Models

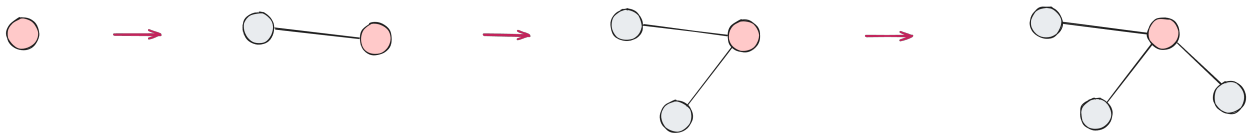

Figure 2.3: Autoregressive models gradually build up a molecule atom-by-atom.

Another class of models that has possibly been overshadowed by the recent developments in diffusion and flow-matching models are auto-regressive models. Auto-regressive models model the distribution of a molecule described as a sequence $\{x_i\}_{i=1}^n$ where $x_i \in \mathbb{R}^{d+3}$, of atom positions (3-dimensional) and features (d-dimensional):

$$p(\{x_i\}_{i=1}^n) = p(x_1) \prod_{i=1}^{N} p(x_i \mid x_0, \ldots, x_{i-1}) \tag{2.2}$$

The sampling process is depicted in Figure 2.3, showing ancestral sampling starting from x_1 to generate an entire molecule. Essentially, the model learns how to place the 'next' atom conditioned on all of the previously placed atoms. Of course, it is essential to guarantee the $E(3)$-equivariance property (Equation 2.1), which will be guaranteed if the marginals $p(x_i|x_0, \ldots, x_{i-1})$ are $E(3)$-equivariant:

$$p(\rho(g)x_i \mid \rho(g)x_0, \ldots, \rho(g)x_{i-1}) = p(x_i|x_0, \ldots, x_{i-1}) \quad \text{for all} \quad i \in [n], g \in E(3) \tag{2.3}$$

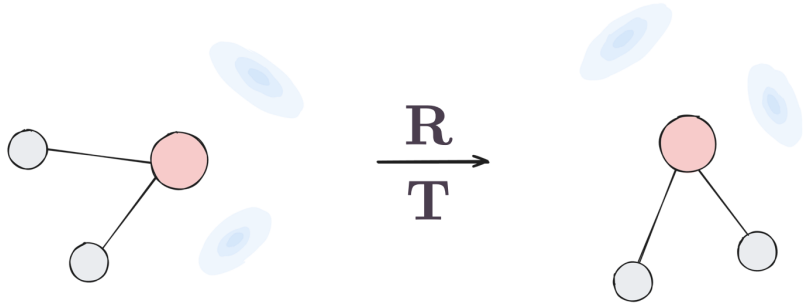

Figure 2.4: $E(3)$-equivariance under rotations $\mathbf{R}$ and translations $\mathbf{T}$ for each marginal distribution $p(x_i|x_0, \ldots, x_{i-1})$ guarantees the $E(3)$-equivariance of $p(\{x_i\}_{i=1}^n)$.

A disadvantage of autoregressive models is that the loss of permutation equivariance, because an order has been imposed on the generated atoms.

One of the first autoregressive model for generating 3D molecular structures was G-SchNet, introduced by Gebauer et al. [15]. To handle the translation equivariance, they first predict a focus atom f amongst already placed atoms.

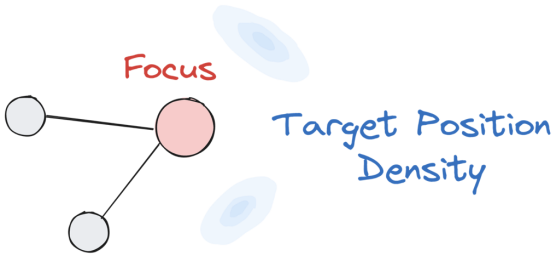

Figure 2.5: Defining the target position relative to a focus atom guarantees translational invariance.

A 3D grid is constructed and queried to predict occupation probabilities for the next atom to be placed. Because Gebauer et al. [15] uses rotationally-invariant features, many atoms need to be queried to compute the probability for each grid point. To break the symmetry when only a few atoms are present in the fragment, auxiliary tokens are added to the fragment. The SchNet [45] $E(3)$-invariant graph neural network is used to learn these probabilities. A conditional variant of the G-SchNet architecture [16] was later developed to condition the generation on desired chemical and physical properties, such as a small HOMO-LUMO gap.

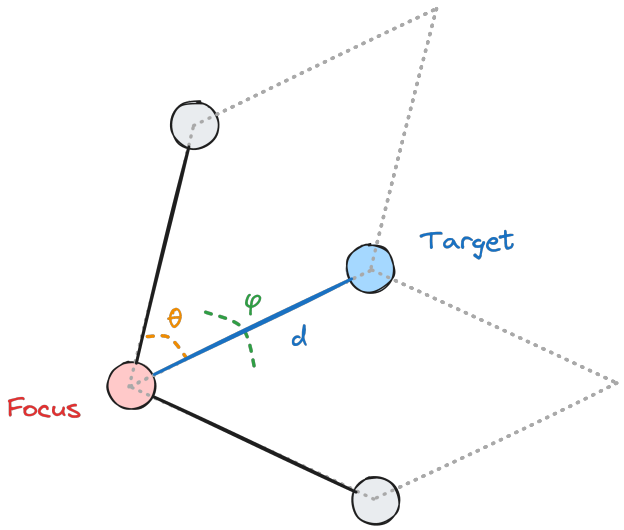

Figure 2.6: The triangulation procedure to place the next atom as depicted in Simm et al. [47] and G-SphereNet [30].

G-SphereNet by Luo and Ji [30] builds off the ideas in G-SchNet by learning an autoregressive flow over atom types. Further, it uses a triangulation procedure (Figure 2.6) to predict the position of the next atom relative to the focal atom and its two nearest neighbors using a distance d, angle θ and dihedral angle φ. The triangulation procedure avoids querying a large dense 3D grid as in G-SchNet. The main drawback of this procedure is its lack of robustness to small deviations in inputs, causing the set of nearest neighbors to change, making the predicted angles meaningless. A similar triangulation procedure was used in Simm et al. [47].

Simm et al. [49] attempts to improve the parametrization of the target position, by using spherical harmonic projections. In particular, the density is expressed in spherical coordinates (r, θ, ϕ) as:

$$p(r \mid f) = \sum_{m=1}^{M} \pi_m(f) \mathcal{N}(\mu_m(f), \sigma_m(f)^2) \tag{2.4}$$

$$p(\theta, \phi \mid r, f) = \frac{1}{Z} \exp\left(-\frac{\beta}{\sqrt{k}} \left| \sum_{l=1}^{l_{\max}} \sum_{m=-l}^{l} c_{lm}(r, f) Y_{lm}(\theta, \phi) \right|^2 \right) \tag{2.5}$$

where f is the focus atom and $k = \sum_{l=1}^{l_{\max}} \sum_{m=-l}^{l} |c_{lm}(f)|^2$ is chosen to regularize the distribution away from a delta distribution. To guarantee $E(3)$-equivariance, the coefficients c_{lm} need to be $E(3)$-equivariant:

$$c_{lm}(\rho(R)f) = \sum_{m'=-l}^{l} D^l_{mm'}(R) c_{lm'}(f)$$

Simm et al. [49] predicts these coefficients with a E(3)-equivariant graph neural network, Cormorant [2]. However, similar to [47], their setting is quite different from the setting of G-SchNet and G-SphereNet. They assume access to a 'bag' of atoms to be placed, and use a reinforcement learning algorithm – Proximal Policy Optimization [44] – to learn a policy to place atoms. The

additional complexity of learning only by sampling possible actions restricts their approach from scaling up to larger datasets with more interesting molecules. Further, [49] proposes rejection sampling to sample from the angular probability distribution defined in Equation 2.5, which is quite inefficient. Finally, an issue is that the angular frequency of the learned probability distribution is strictly bounded by the cutoff l_{max} of the spherical harmonics.

Chapter 3

Background

3.1 $E(3)$-Equivariant Features

We say a $E(3)$-equivariant feature $z = f(x) \in \mathbb{R}^{2l+1}$ transforms as the irreducible representation l (ie, as a l 'irrep') under rotation $\mathbf{R}$ and translation $\mathbf{T}$:

$$f(\mathbf{R}x + \mathbf{T}) = D^l(\mathbf{R})^T f(x) \tag{3.1}$$

where D^l is the irreducible representation of $SO(3)$ of degree $2l + 1$. $D^l(\mathbf{R}) \in \mathbb{R}^{(2l+1)\times(2l+1)}$ is referred to as the Wigner D-matrix of the rotation $\mathbf{R}$. As $D^0(\mathbf{R}) = 1$ and $D^1(\mathbf{R}) = \mathbf{R}$, invariant 'scalar' features correspond to degree $l = 0$ features, while 'vector' features correspond to $l = 1$ features. Note that these features are invariant under translation $\mathbf{T}$.

3.2 Spherical Harmonics

The real spherical harmonics $Y_{l,m}(\theta, \phi)$ are a set of real-valued orthogonal functions defined on the sphere S^2, indexed by two integers l and m such that $l \geq 0, |m| \leq l$. Here θ and ϕ come from the notation for spherical coordinates, where r is the distance from an origin, $\theta \in [0, \pi]$ is the polar angle and $\phi \in [0, 2\pi)$ is the azimuthal angle. The relation between Cartesian and spherical coordinates is given by:

$$\begin{aligned} x &= r \sin\theta \cos\phi \\ y &= r \sin\theta \sin\phi \\ z &= r \cos\theta \end{aligned}$$

The first few spherical harmonics $Y_{l,m}(\theta, \phi)$ are shown in Figure 3.1 below. l corresponds to an angular frequency: the higher the l, the more rapidly $Y_{l,m}$ changes over S^2.

l corresponds to an angular frequency: the higher the l, the more rapidly $Y_{l,m}$ changes over S^2. This can intuitively be seen by looking at the functional form of the spherical harmonics. In their Cartesian form, the spherical harmonics are proportional to simple polynomials. In one common choice of basis, $l = 0$ is proportional to 1, $l = 1$ is proportional to (x, y, z) and $l = 2$ is proportional to $(xy, yz, 2z^2 - x^2 - y^2, zx, x^2 - y^2)$, as seen in Figure 3.1.

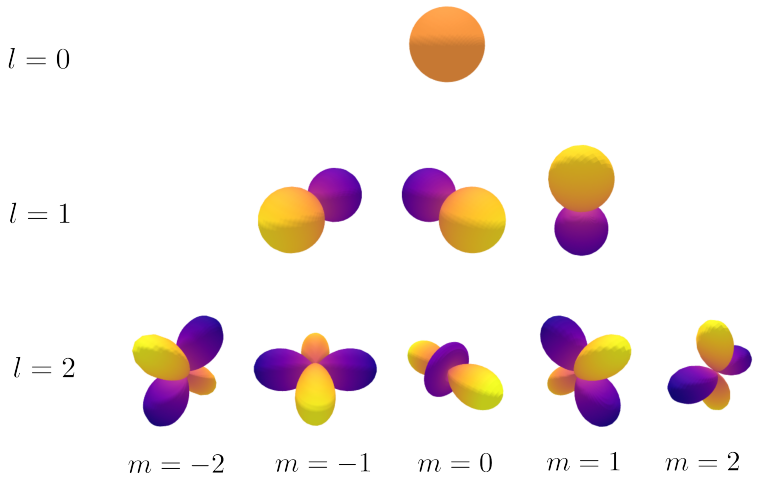

Figure 3.1: Plots of $Y_{l,m}(\theta, \phi)$ as l varies from 0 to 2, and m varies from $-l$ to l. The radial component and color intensity is proportional to the amplitude of $Y_{l,m}(\theta, \phi)$ at each point $(1, \theta, \phi)$.

One important property of the spherical harmonics is that they can be used to create $E(3)$-*equivariant* features. Let $Y_l(\theta, \phi) = [Y_{l,-l}(\theta, \phi), \ldots, Y_{l,l}(\theta, \phi)] \in \mathbb{R}^{2l+1}$ represent the collection of all spherical harmonics with the same l. Then, $Y_l(\theta, \phi)$ transforms as an $E(3)$-equivariant feature of degree l under rotation:

$$Y_l(\mathbf{R} \cdot (\theta, \phi)) = D^l(\mathbf{R})^T Y_l(\theta, \phi) \tag{3.2}$$

where $\mathbf{R}$ is an arbitrary rotation, and (θ, ϕ) is interpreted as the coordinates of a point on S^2.

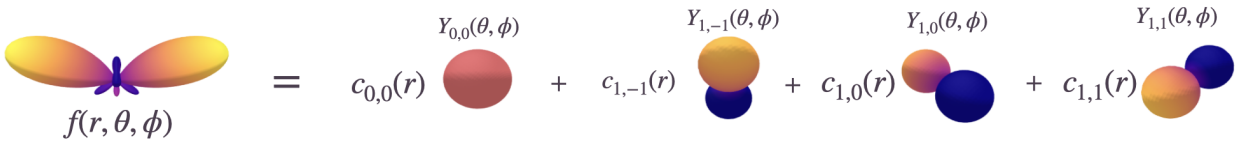

Figure 3.2: Decomposing a function into a linear combination of spherical harmonics.

The second important property of the spherical harmonics that we employ is the fact that they form an *orthonormal basis* for functions on the sphere S^2. Thus, for any function $f : S^2 \to \mathbb{R}$, we can express f as a linear combination of the $Y_{l,m}$. Formally, there exists unique coefficients $c_l \in \mathbb{R}^{2l+1}$ for each $l \in \mathbb{N}$, such that

$$f(\theta, \phi) = \sum_{l=0}^{\infty} {c_l}^T Y_l(\theta, \phi). \tag{3.3}$$

We term these coefficients c_l as the spherical harmonic coefficients of f as they are obtained by projecting f onto the spherical harmonics.

Chapter 4

Methods

4.1 Building Molecules via Sequences of Fragments

First, we create sequences of fragments using molecules from the training set via CreateFragmentSequence. Given a molecule M and random seed r, CreateFragmentSequence constructs a sequence of increasingly larger fragments $\{\mathcal{S}^1, \ldots \mathcal{S}^{|M|}\}$ such that $|\mathcal{S}^n| = n$ for all $n \in \{1, \ldots, |M|\}$ and $\mathcal{S}^{|M|} = M$ exactly. Of course, there are many ways to create such sequences of fragments; CreateFragmentSequence simply builds a molecule via a minimum spanning tree.

Symphony attempts to recreate this sequence step-by-step, learning the (probabilistic) mapping $\mathcal{S}^n \rightarrow \mathcal{S}^{n+1}$. In particular, we ask Symphony to predict the focus node f_{n+1}, the target atomic number Z_{n+1} and the target position $\vec{\mathbf{r}}_{n+1}$, providing feedback at every step.

Algorithm 1 CreateFragmentSequence

Input: Molecule M, PRNG Seed s
  Sample an atom $(\vec{\mathbf{r}}_1, Z_1)$ from M using the PRNG seed s.
  $\mathcal{S}^1 \leftarrow \{(\vec{\mathbf{r}}_1, Z_1)\}$
  function NextFragment($\mathcal{S}^n, s$)
    $(f, a) \leftarrow$ Closest atom pair s.t. $f \in \mathcal{S}^n$ and $a \in M - \mathcal{S}^n$
    (Ties are broken randomly using seed s)
    $f_{n+1} \leftarrow$ The index of the atom f in $\mathcal{S}^n$
    $Z_{n+1} \leftarrow$ The atomic number of atom a
    $\vec{\mathbf{r}}_{n+1} \leftarrow$ The relative position of atom a w.r.t. atom f
    $\mathcal{S}^{n+1} \leftarrow \mathcal{S}^n \cup \{(\vec{\mathbf{r}}_{n+1}, Z_{n+1})\}$
    $s' \leftarrow$ Update PRNG Seed s
    return $(\mathcal{S}^{n+1}, s')$
  for $n \leftarrow 1$ to $|M| - 1$ **do**
    $(\mathcal{S}^{n+1}, s) \leftarrow$ NextFragment($\mathcal{S}^n, s$)
  return $\{\mathcal{S}^1, \ldots \mathcal{S}^{|M|}\}$

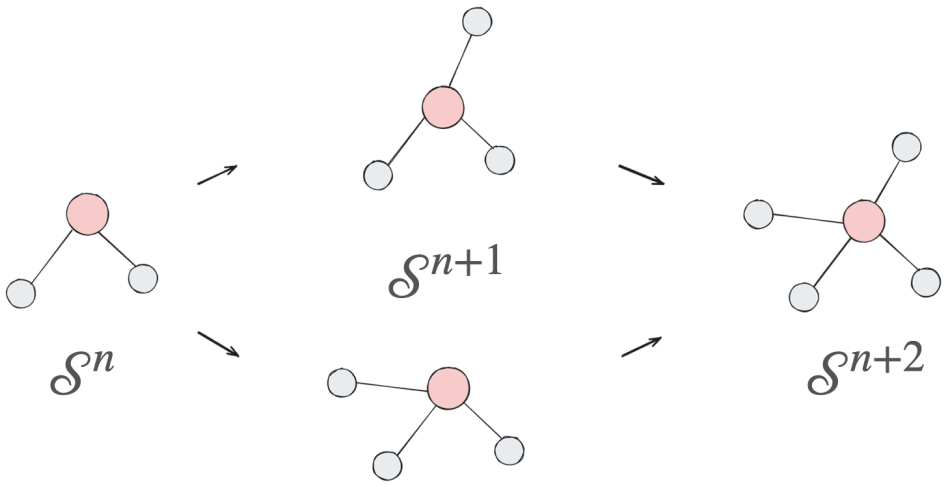

Figure 4.1: Fragments from CreateFragmentSequence applied to methane (CH4). Note the two options available to complete the initial fragment $\mathcal{S}^n$.

4.2 Handling the Symmetries of Fragments

Here, we highlight several challenges that arise because $\mathcal{S}^n$ must be treated as an unordered set of atoms that live in 3D space. In particular, let

$$\mathbf{R}\mathcal{S}^n + \mathbf{T} = \{(\mathbf{R}\vec{\mathbf{r}}_1 + \mathbf{T}, Z_1), \ldots, (\mathbf{R}\vec{\mathbf{r}}_n + \mathbf{T}, Z_n)\}$$

be the description of the same set of atoms in $\mathcal{S}^n$ but in a coordinate frame rotated by $\mathbf{R}^{-1}$ and translated by $\mathbf{T}^{-1}$. Similarly, let π be any permutation of $\{1, \ldots, n\}$ and

$$\pi\mathcal{S}^n = \{(\vec{\mathbf{r}}_{\pi(1)}, Z_{\pi(1)}), \ldots, (\vec{\mathbf{r}}_{\pi(n)}, Z_{\pi(n)})\}$$

. Fundamentally, $\mathbf{R}\mathcal{S}^n + \mathbf{T}$, $\mathcal{S}^n$ and $\pi\mathcal{S}^n$ all represent the same set of atoms. Thus, we would like Symphony to naturally accommodate the symmetries of fragment $\mathcal{S}^n$, under the group $E(3)$ of Euclidean transformations consisting of all rotations $\mathbf{R}$ and translations $\mathbf{T}$, and the group of all permutations of constituent atoms. Formally, we wish to have:

- **Property (1)**: The focus distribution p^{focus} and the target species distribution p^{species} should be $E(3)$-*invariant*:

$$p^{\text{focus}}(f_{n+1}; \mathbf{R}\mathcal{S}^n + \mathbf{T}) = p^{\text{focus}}(f_{n+1}; \mathcal{S}^n) \tag{4.1}$$

$$p^{\text{species}}(Z_{n+1} \mid f_{n+1}; \mathbf{R}\mathcal{S}^n + \mathbf{T}) = p^{\text{species}}(Z_{n+1} \mid f_{n+1}; \mathcal{S}^n) \tag{4.2}$$

- **Property (2)**: The target position distribution p^{pos} should be $E(3)$-*equivariant*:

$$p^{\text{pos}}(\mathbf{R}\vec{\mathbf{r}}_{n+1} + \mathbf{T} \mid f_{n+1}, Z_{n+1}; \mathbf{R}\mathcal{S}^n + \mathbf{T}) = p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n) \tag{4.3}$$

- **Property (3)**: With respect to the ordering of atoms in $\mathcal{S}^n$, the map p^{focus} should be permutation-equivariant while p^{species} and p^{pos} should be permutation-invariant.

We represent p^{focus}, p^{species} and p^{pos} as probability distributions because there may be multiple valid choices of focus f_{n+1}, species Z_{n+1} and position $\vec{\mathbf{r}}_{n+1}$.

4.3 The Design of Symphony

The overall working of Symphony is shown graphically in Figure 4.2. Symphony first computes atom embeddings via an $E(3)$-equivariant EMBEDDER. Here, we assume that EMBEDDER$(\mathcal{S}^n) = \{h_{v,l} \mid v \in \mathcal{S}^n, 0 \leq l \leq l_{\max}\}$ returns a set of $E(3)$-equivariant features $h_{v,l}$ of degree l upto a predefined degree $l_{\max}$, for each atom v in $\mathcal{S}^n$. In Appendix A, we show that Symphony can guarantee Properties (1), (2) and (3) as long as EMBEDDER is permutation-equivariant and $E(3)$-equivariant.

Figure 4.2: One iteration of the Symphony molecular generation process, in which one atom is sampled given the positions and atom types of an unfinished molecular fragment $\mathcal{S}^n$ to create the next fragment $\mathcal{S}^{n+1}$.

From Property (1), p^{focus} and p^{species} should be invariant under rotation and translations of $\mathcal{S}^n$. Since the atom types and the atom indices are discrete sets, we can represent both of these distributions as a vector of probabilities. Thus, we compute p^{focus} and p^{species} by applying a multi-layer perceptron MLP on only the rotation and translation invariant features of EMBEDDER$(\mathcal{S}^n)$:

$$p^{\text{focus}}(f_{n+1}; \mathcal{S}^n) = \text{MLP}(\text{EMBEDDER}(\mathcal{S}^n)_{f_{n+1},0}) \tag{4.4}$$

$$p^{\text{species}}(Z_{n+1} \mid f_{n+1}; \mathcal{S}^n) = \text{MLP}(\text{EMBEDATOMTYPE}(Z_{n+1}) \cdot \text{EMBEDDER}(\mathcal{S}^n)_{f_{n+1},0}) \tag{4.5}$$

Alongside the node-wise probabilities for p^{focus}, we also predict a global STOP probability, indicating that no atom should be added.

On the other hand, Property (2) shows that p^{pos} transforms non-identically under rotations and translations. We describe a novel parametrization of 3D probability densities such as p^{pos} with spherical harmonic projections.

The position $\vec{\mathbf{r}}$ is represented by spherical coordinates (r, θ, ϕ) where r is the distance from the focus f, θ is the polar angle and ϕ is the azimuthal angle. Any probability distribution p^{pos} over positions must satisfy the normalization and non-negativity constraints:

$$\int_\Omega p^{\text{pos}}(r, \theta, \phi)\, dV = 1 \tag{4.6}$$

$$p^{\text{pos}}(r, \theta, \phi) \geq 0 \tag{4.7}$$

where $dV = r dr \sin\theta d\theta d\phi$ is the volume element and $\Omega = \{r \in [0, \infty), \theta \in [0, \pi], \phi \in [0, 2\pi)\}$ represents all space in spherical coordinates. Since these constraints are hard to incorporate directly into a neural network, we predict the unnormalized logits $f^{\text{pos}}(r, \theta, \phi)$ instead, and take the softmax over all space:

$$p^{\text{pos}}(r, \theta, \phi) = \frac{1}{Z} \exp f^{\text{pos}}(r, \theta, \phi) \tag{4.8}$$

To model these logits, we first discretize the radial component r into a set of discrete values. We choose 64 uniformly spaced values from 0.9A to 2.0A, which covers all of the bond lengths in QM9. For each fixed value of r, we obtain a function on the sphere S^2, which we represent in the basis of spherical harmonic functions $Y_{l,m}(\theta, \phi)$, as described in chapter 3 and similar to the construction of Cohen and Welling [11]. As we have a radial component r here, the coefficients c_l also depend on r:

$$f^{\text{pos}}(r, \theta, \phi \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n) = \sum_{l=0}^{\infty} c_l(r; f_{n+1}, Z_{n+1}, \mathcal{S}^n)^T Y_l(\theta, \phi) \tag{4.9}$$

Symphony predicts these coefficients c_l from the degree l features of the focus node $\textsc{Embedder}(\mathcal{S}^n)_{f_{n+1},l}$, and the embedding of the target species Z_{n+1}:

$$c_l(r; f_{n+1}, Z_{n+1}, \mathcal{S}^n) = \textsc{Linear}(\textsc{Embedder}(\mathcal{S}^n)_{f_{n+1},l} \otimes \textsc{EmbedAtomType}(Z_{n+1})) \tag{4.10}$$

By explicitly modelling the probability distributions p^{focus}, p^{species} and p^{pos}, Symphony learns to represent all possible options of completing $\mathcal{S}^n$ into a valid molecule.

4.4 Bypassing the Angular Frequency Bottleneck

For computational reasons, we are often limited to using a finite number of spherical harmonic projections (ie, up to some l_{max}). Due to the way the spherical harmonics are constructed, this means we can only represent signals upto some angular frequency. For example, to represent a signal on the sphere with peaks separated by d radians, we need spherical harmonic projections with $l_{\text{max}} \geq \frac{2\pi}{d}$. This is similar how we cannot represent high frequency components using the only the first few Fourier components. To bypass the bottleneck of angular frequency, we propose using *multiple* channels of spherical harmonic projections, which are then summed over after a non-linearity:

$$f^{\text{pos}}(r, \theta, \phi; \mathcal{S}^n) = \log \sum_{\text{channel ch}} \exp \sum_{l=0}^{\infty} c_l^{\text{ch}}(r; \mathcal{S}^n)^T Y_l(\theta, \phi) \tag{4.11}$$

In Appendix B, we show a concrete example where adding multiple channels effectively increases the angular frequency capacity of our model.

4.5 Training and Inference

We utilize teacher forcing to train Symphony. At training time, the true focus f_{n+1} and atomic number Z_{n+1} are provided as computed in NextFragment. Thus, no sampling occurs at training time. The true probability distributions q^{focus} and q^{species} are computed empirically from the set of unfinished atoms in $\mathcal{S}^n$ and their corresponding neighbors in M. The true probability distribution q^{position} is computed by smoothly approximating a Dirac delta distribution upto some cutoff frequency l_{max} at the target position $\vec{\mathbf{r}}_{n+1}$ around the focus atom. Further details about

the training process and representing Dirac delta distributions are provided in Section E.1.2 and Section C.3.

$$q^{\text{pos}}(\vec{\mathbf{r}}) = \frac{1}{Z} \exp\left(-\frac{\|\vec{\mathbf{r}}\| - \|\vec{\mathbf{r}}_{n+1}\|}{2\sigma^2_{\text{true}}} \cdot \delta_{l_{\max}}\left(\hat{\mathbf{r}} - \hat{\mathbf{r}}_{n+1}\right) \right) \tag{4.12}$$

At inference time, both the focus f_{n+1} and atomic number Z_{n+1} are sampled from $p^{\text{focus}}(\cdot; \mathcal{S}^n)$ and $p^{\text{species}}(\cdot \mid f_{n+1}; \mathcal{S}^n)$ respectively. These are used to sample $\vec{\mathbf{r}}_{n+1}$ from $p^{\text{pos}}(\cdot \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n)$. Molecules are generated by starting from an initial fragment $\mathcal{S}^1$, a single H atom at the origin, and repeatedly sampling from p^{focus}, p^{species} and p^{pos} until a STOP is predicted or $N_{\max} = 35$ iterations have occurred, based on the maximum size of molecules in the QM9 dataset as 30 atoms.

Chapter 5

Experimental Results

A major challenge with generative modelling is evaluating the quality of generated 3D structures. Ideally, a generative model should generate physically plausible structures, accurately capture training set statistics and generalize well to molecules outside of its training set. We propose a comprehensive set of tests to evaluate Symphony and other generative models along these three aspects.

5.1 Validity of Generated Structures

All of the generative models considered here output a set of atoms with 3D coordinates; bonding information is not generated by the model. Before we can use cheminformatics tools designed for molecules, we need to assign bonds between atoms. Multiple algorithms exist for bond order assignment: `xyz2mol` [24] implemented in RDKit [27], OpenBabel [5] and a simple lookup table based on empirical pairwise distances in organic compounds [20]. Here, we perform the first comparison between these algorithms for evaluating machine-learning generated 3D structures. In Table 5.1, we use each of these algorithms to infer the bonds and create a molecule from generated 3D molecular structure. We declare a molecule as valid if the algorithm could successfully assign bond order with no net resulting charge. We also measure the uniqueness to see how many repetitions were present in the set of SMILES [56] strings of valid generated molecules. Ideally, we want both the validity and the uniqueness to be high.

While EDM [20] is still superior on the validity and uniqueness metrics, we find that Symphony performs much better on both validity and uniqueness than existing autoregressive models, G-SchNet [15] and G-SphereNet [30], for the `xyz2mol` and OpenBabel algorithms. Note that the lookup table does not account for aromatic bonds and is quite sensitive to exact bond lengths; we believe this penalizes Symphony due to its coarser discretization compared to EDM and G-SchNet. Of note is that only `xyz2mol` finds almost all of the ground truth QM9 structures to be valid.

Recently, Buttenschoen et al. [10] showed that the predicted 3D structures from machine-learned protein-ligand docking models tend to be highly unphysical. For Table 5.2, we utilize their PoseBusters framework to perform the following sanity checks to count how many of the predicted 3D structures are reasonable. We see that the valid molecules from all models tend to be quite reasonable, with Symphony performing better than all baselines on generating struc-

Metric ↑	QM9	Symphony	EDM	G-SchNet	G-SphereNet
Validity via `xyz2mol`	99.99	83.50	**86.74**	74.97	26.92
Validity via OpenBabel	94.60	74.69	**77.75**	61.83	9.86
Validity via Lookup Table	97.60	68.11	**90.77**	80.13	16.36
Uniqueness via `xyz2mol`	99.84	97.98	**99.16**	96.73	21.69
Uniqueness via OpenBabel	99.97	99.61	**99.95**	98.71	7.51
Uniqueness via Lookup Table	99.89	97.68	**98.64**	93.20	23.29

Table 5.1: Validity and uniqueness (among valid) percentages of molecules with different bond assignment methods, with **best** and second-best models highlighted.

tures with reasonable UFF [36] energies and respecting the geometry constraints of double bonds. Further details about the PoseBusters tests are provided in Section E.2.1.

Test ↑	Symphony	EDM	G-SchNet	G-SphereNet
All Atoms Connected	99.92	99.88	99.87	**100.00**
Reasonable Bond Angles	99.56	**99.98**	99.88	97.59
Reasonable Bond Lengths	98.72	**100.00**	99.93	72.99
Aromatic Ring Flatness	**100.00**	**100.00**	99.95	99.85
Double Bond Flatness	**99.07**	98.58	97.96	95.99
Reasonable Internal Energy	**95.65**	94.88	95.04	36.07
No Internal Steric Clash	98.16	**99.79**	99.57	98.07

Table 5.2: Percentage of valid (as obtained from `xyz2mol`) molecules passing each PoseBusters test.

5.2 Capturing Training Set Statistics

Next, we evaluate models on how well they capture bonding patterns and the geometry of local environments found in the training set molecules. In previous work [20, 30], models were compared based on how well they capture the true bond length distributions observed in QM9. However, such statistics only deal with pairwise bond lengths and cannot capture the geometry of how atoms are placed relative to each other. Here, we utilize the *bispectrum* [52] as a rotationally invariant descriptor of the geometry of local environments. Given a local environment with a central atom u, we first project all of the neighbors of u according to the inferred bonds onto the unit sphere S^2. Then, we compute the signal f as a sum of Dirac delta distributions along the direction of each neighbor: $f(\hat{\mathbf{r}}) = \sum_{v \in N(u)} \delta_{l_{\max}}(\hat{\mathbf{r}} - \hat{\mathbf{r}}_{vu})$. The bispectrum $\mathcal{B}(f)$ of f is then defined as:

$$\mathcal{B}(f) = \textsc{ExtractScalars}(f \otimes f \otimes f) \tag{5.1}$$

Thus, f captures the distribution of atoms around u, and the bispectrum $\mathcal{B}(f)$ captures the geometry of this distribution. The advantage of the bispectrum is that it varies smoothly when

f is varied and is guaranteed to be rotationally invariant. We compute the bispectrum of local environments with atleast 2 neighboring atoms. Note that we exclude the pseudoscalars in the bispectra.

For comparing discrete distributions, we use the symmetric Jensen-Shannon divergence (JSD) as employed in Hoogeboom et al. [20]. Given the true distribution Q and the predicted distribution P, the Jensen-Shannon divergence between them is defined as:

$$D_{JS}(Q \parallel P) = \frac{1}{2} D_{KL}\left(Q \parallel M\right) + \frac{1}{2} D_{KL}\left(P \parallel M\right) \tag{5.2}$$

where D_{KL} is the Kullback–Leibler divergence and $M = \frac{Q+P}{2}$ is the mean distribution. For continuous distributions, estimating the Jensen-Shannon divergence from samples is tricky without further assumptions on the distributions. Instead, we use the Maximum Mean Discrepancy (MMD) score from Luo and Ji [30] instead to compare samples from continuous distributions. The MMD score is the distance between means of features computed from samples from the true distribution Q and the predicted distribution P. A model with a smaller MMD score captures the true distribution of samples better. We provide details about the MMD score in Section E.2.2.

From Table 5.3 we see that Symphony and other autoregressive models struggle to match the bond length distribution of QM9 as well as EDM. This is the case except for the single C-H and single N-H bonds. On the bispectra, however, Symphony attains the lowest MMD for several environments. To gain some intuition for these MMD numbers, we also plotted the bond length distributions, samples of the bispectra, atom type distributions and other statistics in **??** for each model.

5.3 Generalization Capabilities

All of the metrics discussed so far can be maximized by simply memorizing the training set molecules. Now, we propose a new metric to evaluate how well the models have actually learned to generate valid chemical structures. We compare models by asking them to complete fragments of 1000 unseen molecules from the test set, with one hydrogen atom removed. We then check how many final molecules were deemed valid. Since the valid completion rate (VCR) depends heavily on the quality of the model, we compute the valid completion rate for fragments of molecules from the training set as well. If the performance is significantly different between the two sets of fragments, this indicates that the models do not generalize well. Diffusion models such as EDM are more challenging to evaluate for this task, since we would need a way to fix the initial set of atoms, so we compare only Symphony and G-SchNet. Encouragingly, both models are able to generalize well to unseen fragments, but Symphony's overall completion rate is higher for both seen and unseen fragments. We notice that the performance of Symphony on this task seems to decrease as training progresses, the reason for which remains unclear.

5.4 Molecule Generation Throughput

One of the major advantages of autoregressive models (such as Symphony) over diffusion models (such as EDM) is significantly faster inference speeds. As shown in **??**, Symphony is much slower

than existing autoregressive models because of the additional tensor products for generating higher-degree $E(3)$-equivariant features, but is still approximately 3× faster than EDM. However, our sampler is currently bottlenecked by some of the limitations of JAX [9]; we believe that Symphony's inference speed reported here can be significantly improved to match its training speed.

5.5 Statistics of Generated Molecules

For all of the analyses performed in this section, we used all the valid molecules for each model as computed by `xyz2mol`.

5.5.1 Bispectra of Local Environments in Sampled Molecules

As seen in Figure 5.1, we see that Symphony's sampled bispectra (second from left) have a slightly different distribution relative to those from QM9 in the two most frequent local environments.

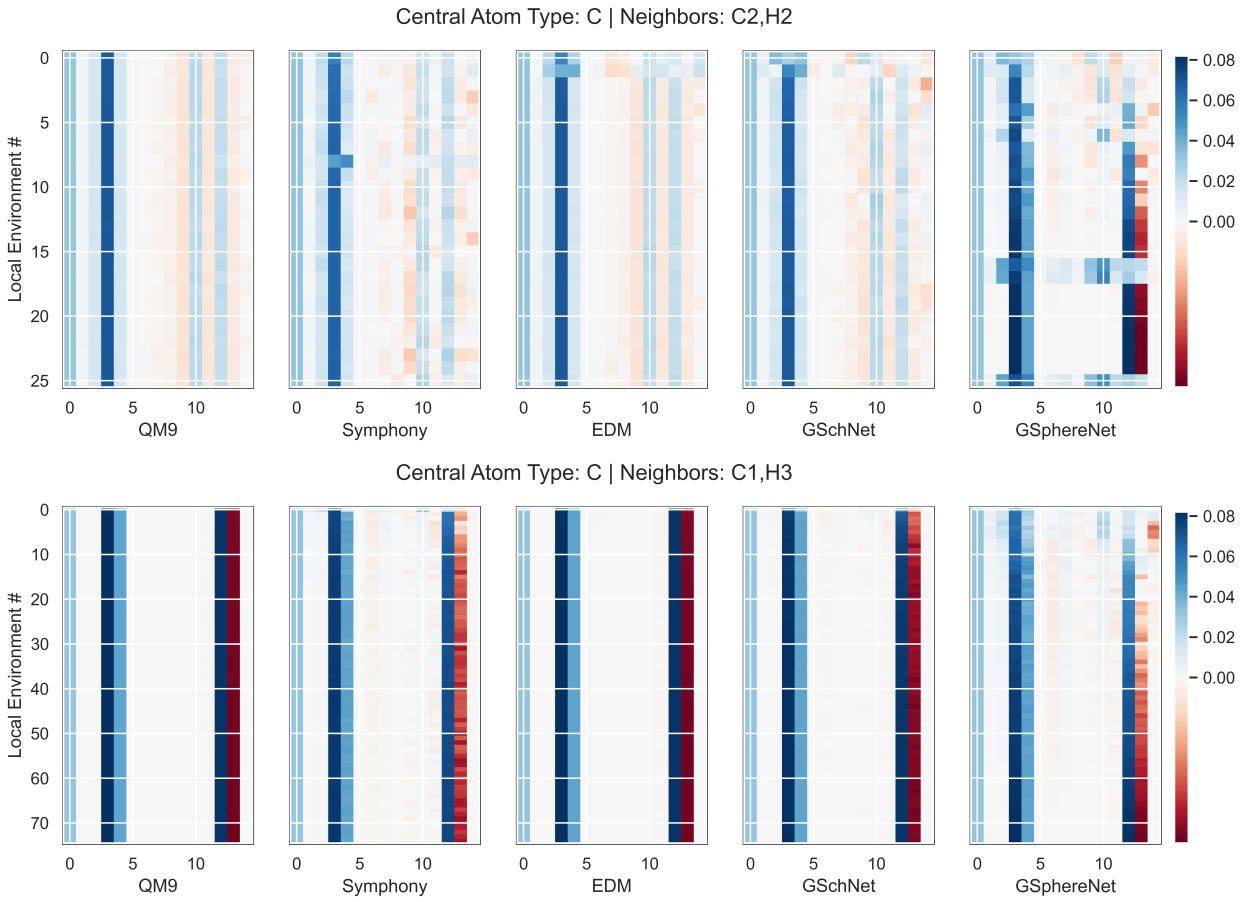

Figure 5.1: Bispectra of local environments of type C: C2,H2 and type C: C1,H3 respectively. Each row corresponds to a sample of the bispectrum (an array of length 15). Every entry of the bispectra is colored by value according to the colorbar on the right.

5.5.2 Bond Lengths in Sampled Molecules

From Figure 5.2 and Figure 5.3, we see that Symphony's bond length distribution tends to be wider than those of QM9, hurting its MMD score relative to EDM. Improving this aspect is an ongoing effort; but we believe that the bond lengths are still quite reasonable.

5.5.3 Atom Type Counts

As seen in Figure 5.4, all models are able to reasonably capture the distribution of atom types in QM9; Symphony performs especially well here.

5.5.4 Ring Sizes

We also extracted all rings using RDKit [27] and counted their relative frequency, in Figure 5.5. G-SphereNet seems to produce either very large or very small rings. The other models seem to capture the distribution of ring sizes well.

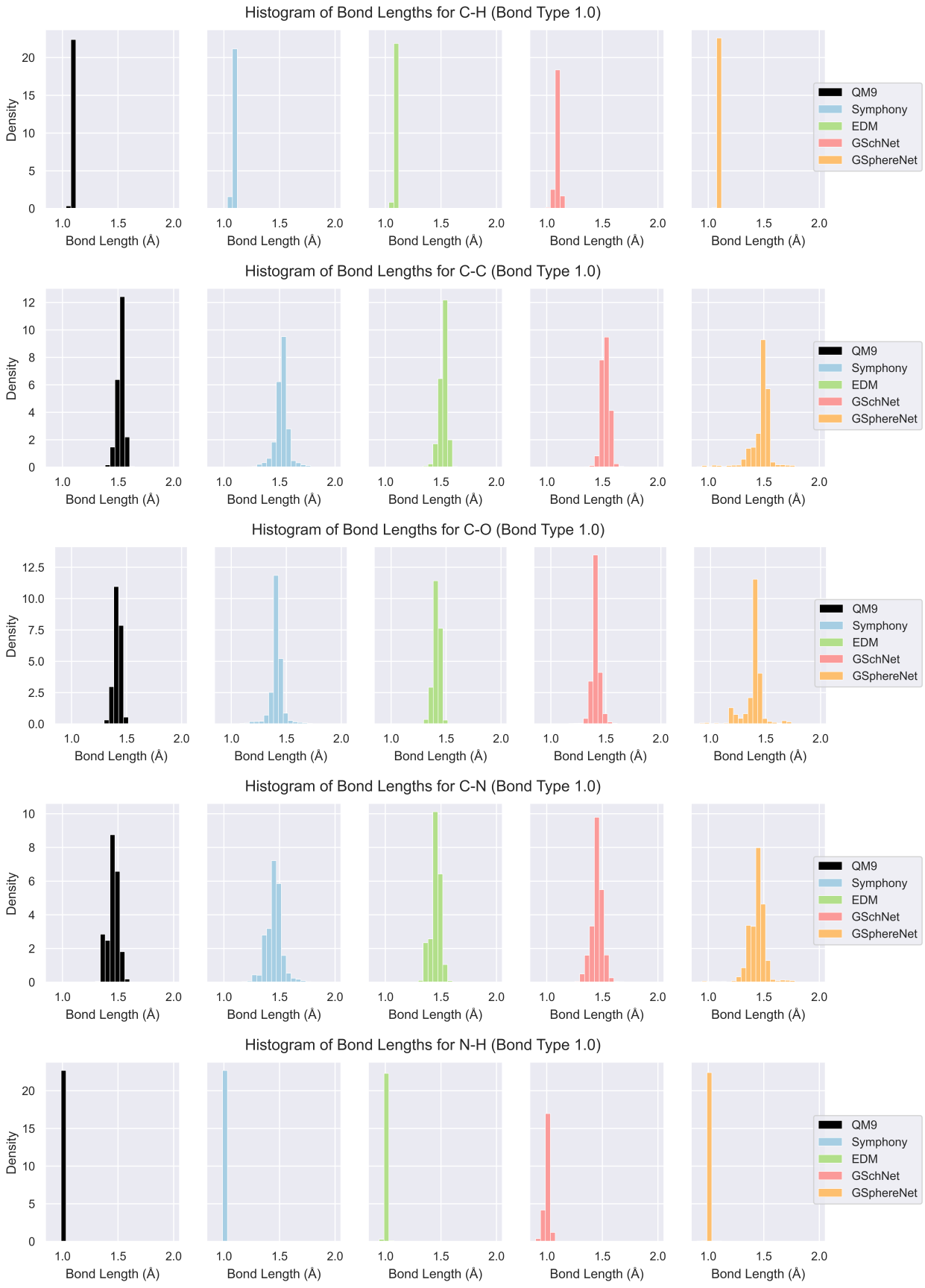

Figure 5.2: Histogram of bond lengths for the five most frequent bonds in QM9.

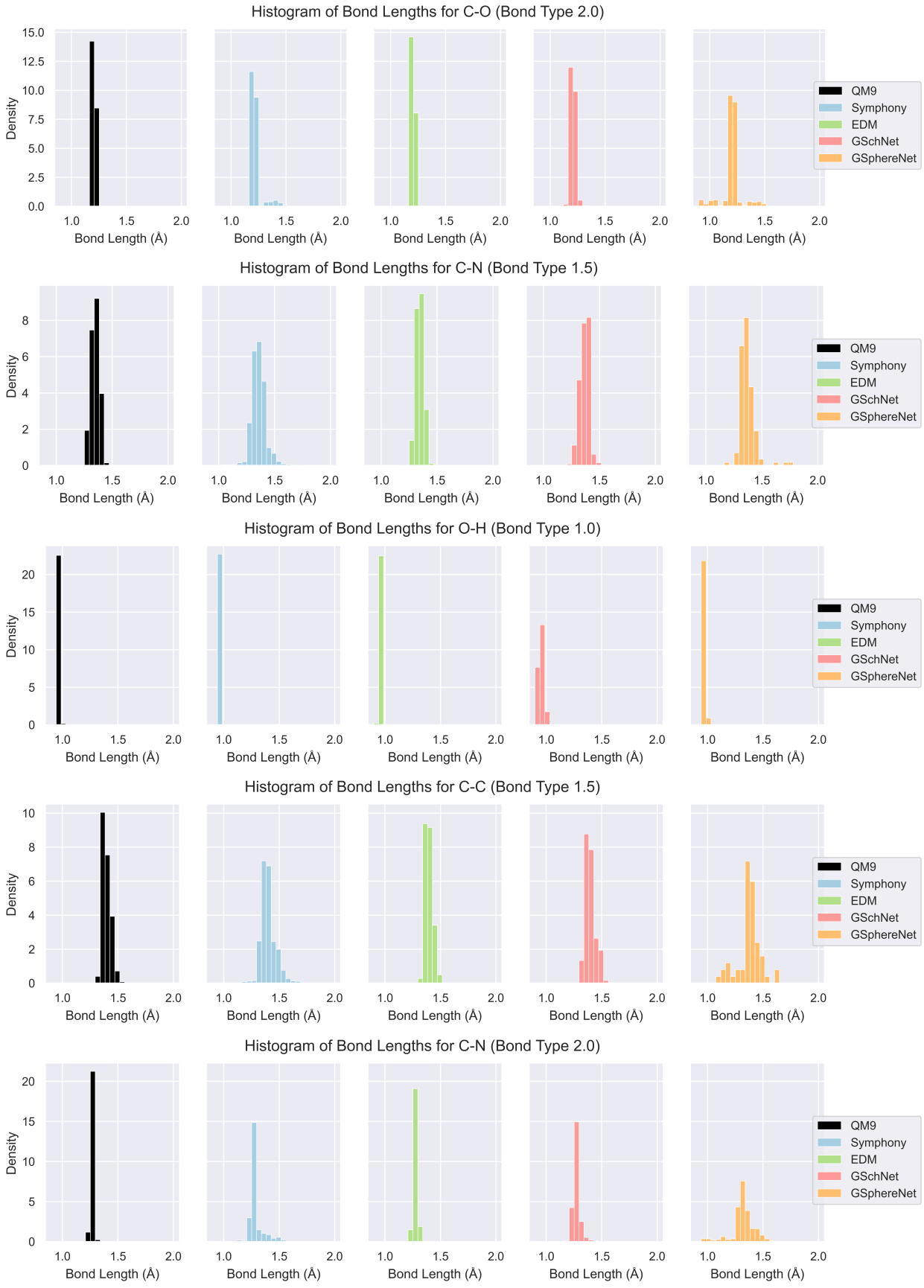

Figure 5.3: Histogram of bond lengths for the sixth to tenth most frequent bonds in QM9.

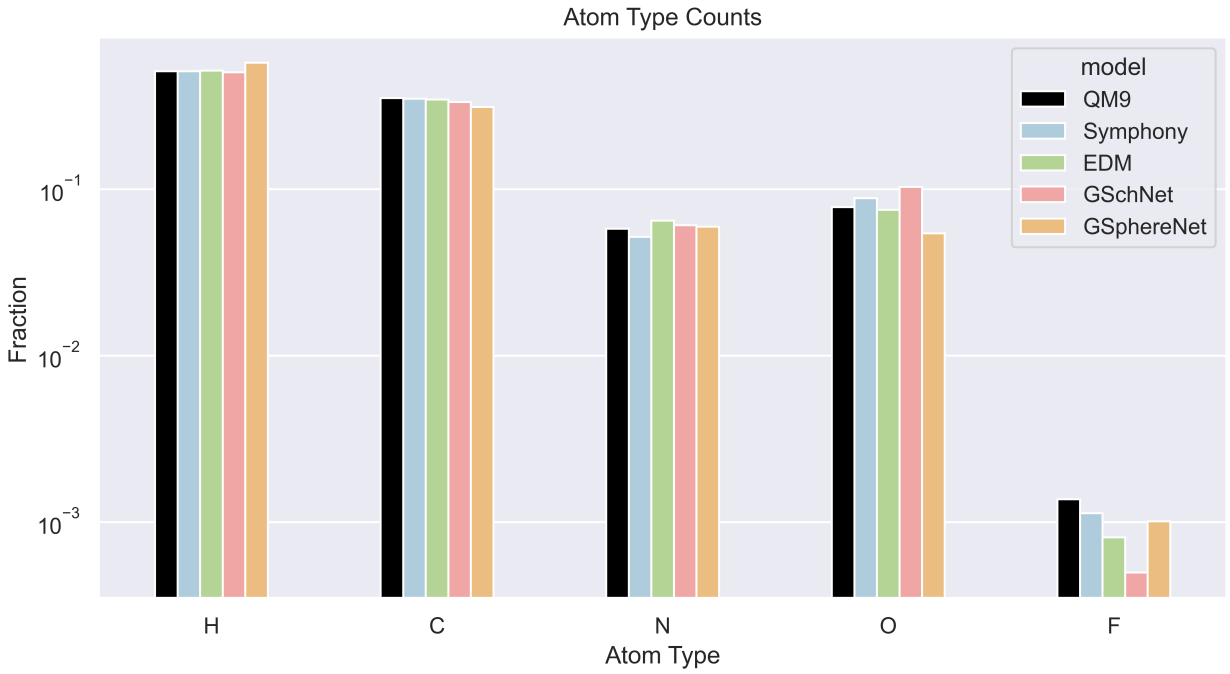

Figure 5.4: Frequency of atom type counts in generated molecules on a log-scale.

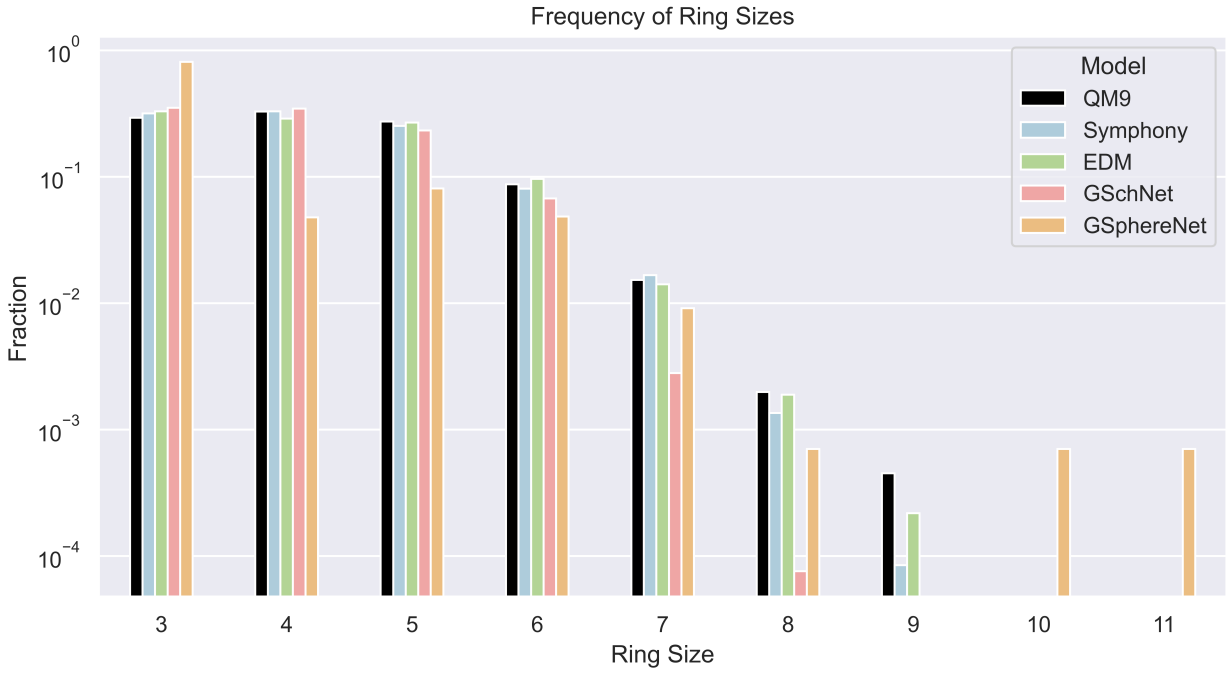

Figure 5.5: Frequency of ring sizes in generated molecules on a log-scale.

MMD of Bond Lengths ↓	Symphony	EDM	G-SchNet	G-SphereNet
C-H: 1.0	0.0739	**0.0653**	0.3817	0.1334
C-C: 1.0	0.3254	**0.0956**	0.2530	1.0503
C-O: 1.0	0.2571	**0.0757**	0.5315	0.6082
C-N: 1.0	0.3086	**0.1755**	0.2999	0.4279
N-H: 1.0	**0.1032**	0.1137	0.5968	0.1660
C-O: 2.0	0.3033	**0.0668**	0.2628	2.0812
C-N: 1.5	0.3707	**0.1736**	0.5828	0.4949
O-H: 1.0	0.2872	0.1545	0.7899	**0.1307**
C-C: 1.5	0.4142	**0.1749**	0.2051	0.8574
C-N: 2.0	0.5938	**0.3237**	0.4194	2.1197

MMD of Bispectra ↓	Symphony	EDM	G-SchNet	G-SphereNet
C: C2,H2	0.2165	**0.1003**	0.4333	0.6210
C: C1,H3	0.2668	**0.0025**	0.0640	1.2004
C: C3,H1	**0.1111**	0.2254	0.2045	1.1209
C: C2,H1,O1	**0.1500**	0.2059	0.1732	0.8361
C: C1,H2,O1	0.3300	0.1082	**0.0954**	1.6772
O: C1,H1	0.0282	0.0056	0.0487	**0.0030**
C: C2,H1,N1	**0.1481**	0.1521	0.1967	1.3461
C: C2,H1	0.2525	**0.0468**	0.1788	0.2403
C: C1,H2,N1	0.3631	0.2728	**0.1610**	0.9171
N: C2,H1	**0.0953**	0.2339	0.2105	0.6141

Jensen-Shannon Divergence ↓	Symphony	EDM	G-SchNet	G-SphereNet
Atom Type Counts	0.0003	**0.0002**	0.0011	0.0026
Local Environment Counts	**0.0039**	0.0057	0.0150	0.1016

Table 5.3: Comparing statistics of generated molecules to those found in QM9. (Top): The MMD of bond lengths for the 10 most frequent bonds. The notation 'X-Y: T' means that a X atom was bonded to a Y atom with a bond of type T. (Middle): The MMD of bispectra for the 10 most occurring local environments. The notation 'X: Yn,Zm' means that an X atom was the central atom, surrounded by n Y atoms and m Z atoms. (Bottom): The JSD of occurrence counts for atom types and local environments. ↓ indicates that lower is better for the metrics.

Valid Completion Rate ↑	Symphony 500K steps	Symphony 800K steps	Symphony 1000K steps	G-SchNet
Training: $\text{VCR}_{\text{train}}$	**98.53**	96.65	95.57	97.91
Testing: VCR_{test}	**98.66**	96.30	95.43	98.15

Table 5.4: Comparing the difference between fragment completion rates on (seen) training and (unseen) testing fragments with one hydrogen removed.

5.6 Visualizing Generated Molecules

Figure 5.6 exhibits random non-cherry-picked samples from Symphony.

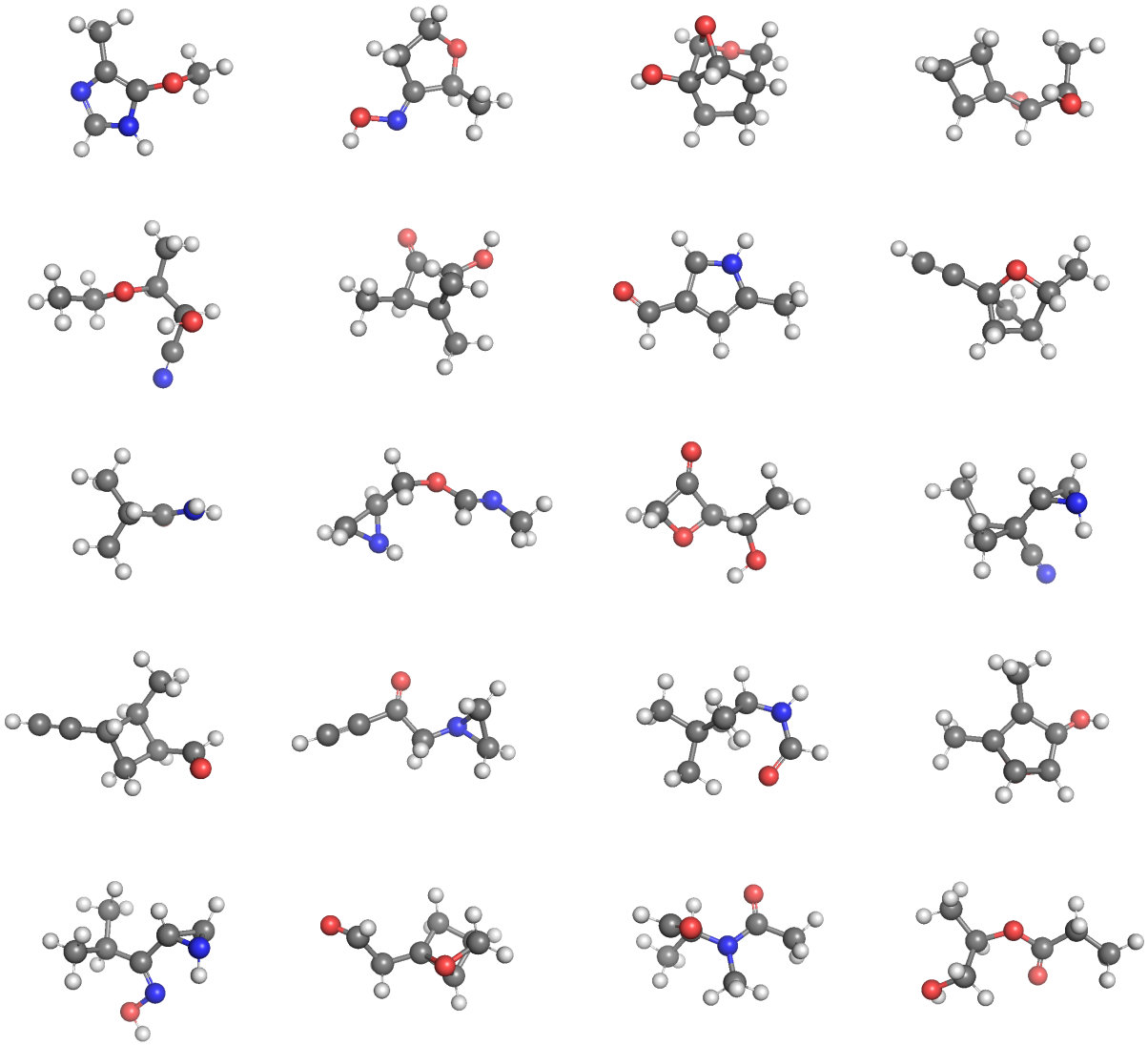

Figure 5.6: Molecules generated by Symphony and visualized with PyMOL [43].

Chapter 6

Conclusion

We have proposed Symphony, a new method to autoregressively generate 3D molecular geometries with spherical harmonic projections and higher-degree $E(3)$-equivariant features. We show promising results on molecular generation and completion, relative to existing autoregressive models. However, one drawback of our current formulation is that the discretization of our radial components is too coarse, so our bond length distributions are not as accurate as EDM or G-SchNet. This affects our validity when using lookup tables to assign bond orders as they are particularly sensitive to exact bond lengths. In the future, we plan to explore training a normalizing flow over radii to avoid this discretization.

Further, Symphony incurs increased computational cost due to the use of tensor products to create higher degree $E(3)$-equivariant features. In the future, we plan to explore normalizing flows to smoothly model the radial distribution without any discretization, and placing entire local environment motifs at once which would speed up generation. On the application side, structure-based drug design such as in [34, 42] to sample binding molecules conditioned on a protein pocket could be a promising task for Symphony.

Understanding how to bridge the gap to the state-of-the-art molecular generation, and actually synthesizing some of these molecules in the lab, are the next steps for me on my research journey.

Appendix A

Proof of Equivariance

A.1 $E(3)$-Equivariance

Theorem: Suppose EMBEDDER produces $O(3)$-equivariant and translation-invariant features $h_{v,l} = \text{EMBEDDER}(\mathcal{S}^n)_{v,l}$ for every atom v. Then, p^{pos} is $O(3)$-equivariant and translation-invariant (and hence, $E(3)$-equivariant):

$$p^{\text{pos}}(\mathbf{R}\vec{\mathbf{r}}_{n+1} + \mathbf{T} \mid f_{n+1}, Z_{n+1}; \mathbf{R}\mathcal{S}^n + \mathbf{T}) = p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n)$$

Proof: We first show that p^{pos} is $O(3)$-equivariant. We have:

$$\text{EMBEDDER}(\mathbf{R}\mathcal{S}^n)_{v,l} = D^l(\mathbf{R})^T \text{EMBEDDER}(\mathcal{S}^n)_{v,l}$$

for every atom v and degree l. Note that because Z_{n+1} is rotationally invariant, it immediately follows from Equation 4.10 and the above, that c_l is also $E(3)$-equivariant with degree l:

$$c_l(r; \mathbf{R}\mathcal{S}^n, f_{n+1}, Z_{n+1}) = c_l(r; \mathcal{S}^n, f_{n+1}, Z_{n+1})$$

Now, as the Wigner D-matrices are always unitary, we have:

$$\begin{aligned}
f^{\text{pos}}(\mathbf{R}\vec{\mathbf{r}}; \mathbf{R}\mathcal{S}^n, f_{n+1}, Z_{n+1}) &= \sum_{l=0}^{\infty} c_l(r; \mathbf{R}\mathcal{S}^n, f_{n+1}, Z_{n+1})^T Y_l(\mathbf{R}\hat{\mathbf{r}}_{ij}) \\
&= \sum_{l=0}^{\infty} c_l(r; \mathcal{S}^n, f_{n+1}, Z_{n+1})^T D^l(\mathbf{R}) D^l(\mathbf{R})^T Y_l(\hat{\mathbf{r}}_{ij}) \\
&= \sum_{l=0}^{\infty} c_l(r; \mathcal{S}^n, f_{n+1}, Z_{n+1})^T Y_l(\hat{\mathbf{r}}_{ij}) \\
&= f^{\text{pos}}(\vec{\mathbf{r}}; \mathcal{S}^n)
\end{aligned}$$

by definition. Thus, we are guaranteed that f^{position} is $O(3)$-equivariant. Note that applying a pointwise non-linearity (exp) to f^{position} and a rotationally invariant normalization does not change $O(3)$-equivariance. Thus, p^{pos} is $O(3)$-equivariant as well.

For translations, note that p^{pos} is described relative to the focus atom f_{n+1}. Thus, as EMBEDDER is translation-invariant:

$$\text{EMBEDDER}(\mathcal{S}^n + \mathbf{T})_{v,l} = \text{EMBEDDER}(\mathcal{S}^n)_{v,l}$$

p^{pos} will be translation-equivariant:

$$p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} + \mathbf{T} \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n + \mathbf{T}) = p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n)$$

In conclusion, p^{pos} is $O(3)$-equivariant and translation-equivariant, and hence $E(3)$-equivariant. Thus, Property (2) is satisfied. ■

A.2 Permutation-Equivariance

Theorem: Suppose EMBEDDER produces permutation-equivariant features $h_{v,l} = \text{EMBEDDER}(\mathcal{S}^n)_{v,l}$ for every atom v. Then, p^{focus} is permutation-equivariant, while p^{species} and p^{pos} are permutation-invariant:

$$\begin{aligned}
p^{\text{focus}}(\pi(f_{n+1}); \pi\mathcal{S}^n) &= p^{\text{focus}}(f_{n+1}; \mathcal{S}^n) \\
p^{\text{species}}(Z_{n+1} \mid \pi(f_{n+1}); \pi\mathcal{S}^n) &= p^{\text{species}}(Z_{n+1} \mid f_{n+1}; \mathcal{S}^n) \\
p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} \mid \pi(f_{n+1}), Z_{n+1}; \pi\mathcal{S}^n) &= p^{\text{pos}}(\vec{\mathbf{r}}_{n+1} \mid f_{n+1}, Z_{n+1}; \mathcal{S}^n)
\end{aligned}$$

where π represents a permutation of the atoms of $\mathcal{S}^n$.
Proof: Because EMBEDDER is permutation-equivariant:

$$\text{EMBEDDER}(\pi\mathcal{S}^n)_{\pi(v),l} = \text{EMBEDDER}(\mathcal{S}^n)_{v,l}$$

for each atom v. Then, from Equation 4.5:

$$\begin{aligned}
p^{\text{focus}}(\pi(f_{n+1}); \pi\mathcal{S}^n) &= \text{MLP}(\text{EMBEDDER}(\pi\mathcal{S}^n)_{\pi(f_{n+1}),0}) \\
&= \text{MLP}(\text{EMBEDDER}(\mathcal{S}^n)_{f_{n+1},0}) \\
&= p^{\text{focus}}(f_{n+1}; \mathcal{S}^n)
\end{aligned}$$

as claimed. Similarly,

$$\begin{aligned}
p^{\text{species}}(Z_{n+1} \mid \pi(f_{n+1}); \pi\mathcal{S}^n) &= \text{MLP}(\text{EMBEDATOMTYPE}(Z_{n+1}) \cdot \text{EMBEDDER}(\pi\mathcal{S}^n)_{\pi(f_{n+1}),0}) \\
&= \text{MLP}(\text{EMBEDATOMTYPE}(Z_{n+1}) \cdot \text{EMBEDDER}(\mathcal{S}^n)_{f_{n+1},0}) \\
&= p^{\text{species}}(Z_{n+1} \mid f_{n+1}; \mathcal{S}^n)
\end{aligned}$$

For p^{pos}, it is sufficient to show that the coefficients $c_l(r)$ are permutation-equivariant:

$$\begin{aligned}
c_l(r; \pi(f_{n+1}), Z_{n+1}, \pi\mathcal{S}^n) &= \text{LINEAR}(\text{EMBEDDER}(\pi\mathcal{S}^n)_{\pi(f_{n+1}),l} \otimes \text{EMBEDATOMTYPE}(Z_{n+1})) \\
&= \text{LINEAR}(\text{EMBEDDER}(\mathcal{S}^n)_{f_{n+1},l} \otimes \text{EMBEDATOMTYPE}(Z_{n+1})) \\
&= c_l(r; f_{n+1}, Z_{n+1}, \mathcal{S}^n)
\end{aligned}$$

Thus, all distributions transform as expected. ■

Appendix B

The Advantage of Using Multiple Channels of Spherical Harmonics

B.1 An Example with the Octahedron

Figure B.1 shows how adding a second channel helps reduce the effective $l_{\max}$ needed to represent p^{pos}. The atoms depicted by red circles have been placed already, and the atom at the center of the octahedron has been chosen as the focus. To accurately capture the positions of the three remaining atoms (depicted by two stars and a square), we would need a projection upto $l_{\max} = 4$, because the angle made by the 'star', central atom and the 'square' is $\frac{\pi}{2}$ radians. However, if we used one channel to represent the 'stars' and one to represent the 'square', we can get away by only using projections upto $l_{\max} = 2$, because the 'stars' are diametrically opposite each other.

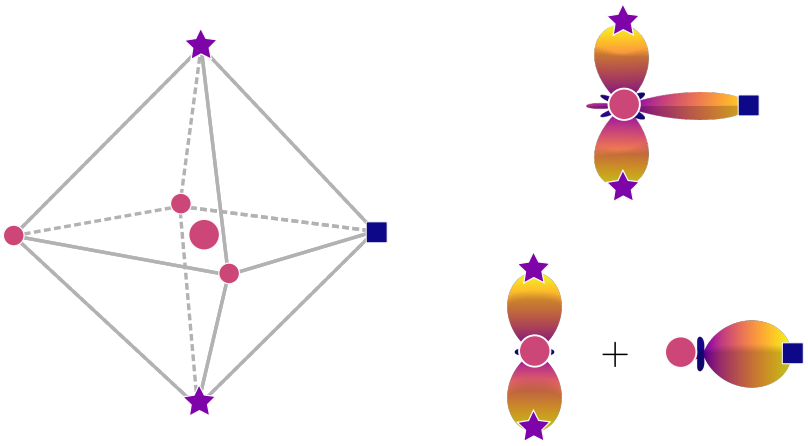

Figure B.1: Usually, we would require $l_{\max} = 4$ to represent p^{pos} for the 'stars' and 'square' atoms, centered at the red central atom. With two channels, we only need up to $l_{\max} = 2$ each.

B.2 A Study on Learning Random Signals

To quantitatively show the effect of having multiple channels, we see how well the model is able to learn a random distribution on the sphere. We randomly sample $N = 5$ target points with coordinates $\{\hat{\mathbf{r}}_i\}_{i=1}^N$ on the sphere, and then define the distribution:

$$q(\hat{\mathbf{r}}) = \sum_{i=1}^{N} \exp(\delta_{l_{\max}}(\hat{\mathbf{r}} - \hat{\mathbf{r}}_i)) \tag{B.1}$$

with the same Dirac delta distribution approximation as described in Section C.3. We use $l_{\max} = 5$ throughout this section. Then, we randomly initialize coefficients c to minimize the KL divergence to q:

$$\min_c KL(q \,||\, p_c) \tag{B.2}$$

where p_c is the probability distribution defined by coefficients c, as before:

$$f(\theta, \phi) = \log \sum_{\text{channel ch}} \exp \sum_{l=0}^{l_{\max}} c_l^{\text{ch}^T} Y_l(\theta, \phi)$$
$$p(\theta, \phi) = \frac{1}{Z} \exp f(\theta, \phi)$$

This corresponds to a simpler setting where we have only one radius r.

We assess the KL divergence as a function of number of position channels ch and $l_{\max}$ in Figure B.2. We see a consistent improvement across different $l_{\max}$ as the number of position channels are increased.

We also experimented with the parametrization from Simm et al. [49], who define:

$$p(\theta, \phi) = \frac{1}{Z} \exp\left(-\frac{\beta}{k} |f(\theta, \phi)|^2\right)$$

where $k = \sum_{l=0}^{l_{\max}} |c_l|^2$. This extra factor of k was proposed by Simm et al. [49] to "regularize the distribution so that it does not approach a delta function". In the left panel of Figure B.3, we show that this regularization hurts the model. Even adding multiple channels does not help, because the regularization term 'switches' off multiples channels. However, as shown in the right panel of Figure B.3, removing this regularization significantly helps the model, with further improvement as the number of channels are increased. For $l_{\max} = 5$, we see that our parametrization performs similarly to Simm et al. [49] without the regularization term. Based on this experiment, we plan to experiment with non-linearities for the logits in future versions of Symphony.

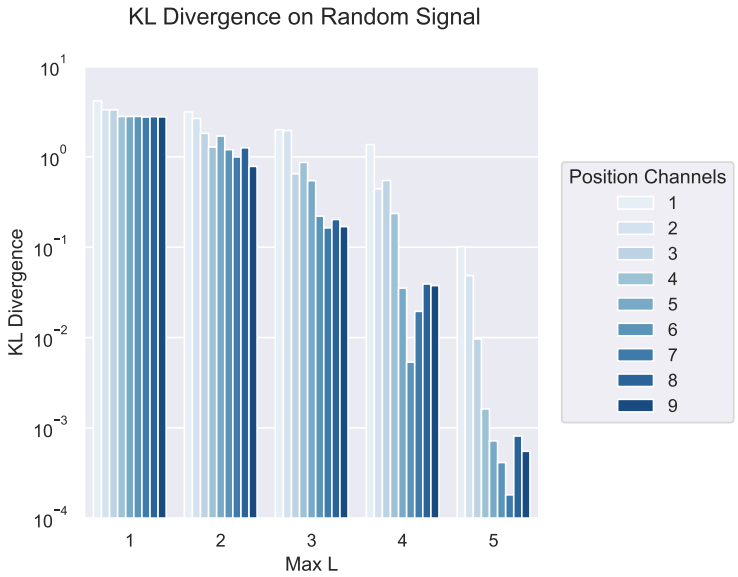

Figure B.2: Final KL divergence $KL(q \mid\mid p_c)$ for learned coefficients c as a function of number of position channels ch and $l_{\max}$.

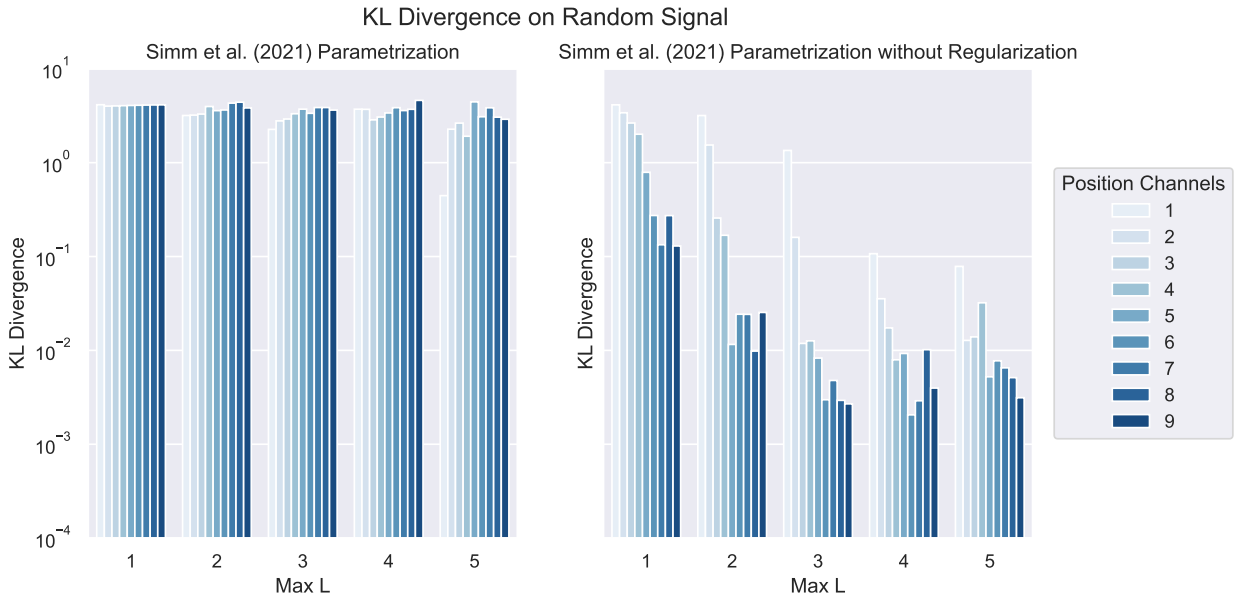

Figure B.3: Final KL divergence $KL(q \mid\mid p_c)$ for learned coefficients c as a function of number of position channels ch and $l_{\max}$, with the parametrization proposed by Simm et al. [49]. Removing the regularization term helps the model learn better.

Appendix C

Learning and Sampling from Position Distributions

In this section, we drop the superscript from p^{pos} as it should be clear from context.

C.1 Learning Spherical Harmonic Coefficients

To recap Section 4.3, Symphony predicts coefficients $c_l^{\text{ch}}(r; \mathcal{S}^n)$ to represent the position distribution p:

$$f(r, \theta, \phi) = \log \sum_{\text{channel ch}} \exp \sum_{l=0}^{\infty} c_l^{\text{ch}}(r; \mathcal{S}^n)^T Y_l(\theta, \phi)$$
$$p(r, \theta, \phi) = \frac{1}{Z} \exp f(r, \theta, \phi)$$

where Z is the partition function.

As mentioned in Section E.1.2, the coefficients are learned by minimizing the KL divergence to the target distribution q:

$$KL(q \mid\mid p) = \int_\Omega q(\vec{\mathbf{r}}) \log \frac{q(\vec{\mathbf{r}})}{p(\vec{\mathbf{r}})} d\vec{\mathbf{r}} = \int_\Omega q(\vec{\mathbf{r}}) \log q(\vec{\mathbf{r}}) d\vec{\mathbf{r}} - \int_\Omega q(\vec{\mathbf{r}}) f(\vec{\mathbf{r}}) d\vec{\mathbf{r}} + \log Z$$

Following the notation of Section 4.3, Ω represents the set $\{r \in [0, \infty), \theta \in [0, \pi], \phi \in [0, 2\pi)\}$ which is all space in spherical coordinates.

For training, we only need the unnormalized logits f and not the normalized distribution p. This is identical to the log-sum-exp trick when training with cross-entropy loss for a classification problem. Unlike the classification case where the number of classes is finite, the integral above must be computed over all of r, θ and ϕ which is an infinite set. To numerically approximate this integral, we use a uniform grid on r and a Spherical Gauss-Legendre quadrature on the sphere at each value of r. As discussed in Section 4.3, the uniform grid on r spans 64 values from 0.9A to 2.0A which is more than sufficient to cover all bond lengths in organic molecules. The Spherical Gauss-Legendre quadrature is a product of two quadratures: a 1D Gauss-Legendre quadrature with 180 points over $\cos\theta \in [-1, 1]$, and a uniform grid of 359 points over $[0, 2\pi)$ for ϕ.

Symphony predicts the coefficients $c_l(r)$ of f which can be used to evaluate $f(r, \theta, \phi)$ at any point. This evaluation for a spherical grid of (θ, ϕ) values can be done quickly via a Fast Fourier Transform (FFT) that is implemented in `e3nn-jax`. We perform this FFT procedure for each sphere defined by a radial grid point r.

C.2 Sampling from the Learned Position Distribution

Once the model is learnt, we need to sample from the distribution p. A key advantage of predicting the coefficients $c_l(r)$ of $f_\theta(r, \theta, \phi)$ is that a different resolution of angular grid can be chosen for sampling than that of training. We simply evaluate $f(r, \theta, \phi)$ on the quadrature grid as before, apply the exponential, and normalize via numerical integration to get $p(r, \theta, \phi)$. We first marginalize over θ, ϕ to obtain a distribution $p(r)$ to sample a radius r. Then, we sample one of the angular grid points (θ, ϕ) for the sphere corresponding to this radius r. Overall, this procedure gives us a sample from $p(r, \theta, \phi)$.

In Section D.2, we assess how the validity of molecules generated by Symphony varies as the grid resolution is varied.

Note that our sampling procedure is much simpler than that of Simm et al. [49], which uses rejection sampling with a uniform base distribution. We perform some quantitative experiments with the parametrization of Simm et al. [49] in Section B.2.

While we are primarily interested in learning distributions over $\mathbb{R}^3$ which are equivariant under $E(3)$, there has been prior work in learning distributions over manifolds [11, 32], where the issue of estimating the partition function are also solved by discretizing over an appropriate domain.

C.3 Representing Dirac Delta Distributions

Suppose we have the function $f(\hat{\mathbf{r}}) = \delta(\hat{\mathbf{r}} - \hat{\mathbf{r}}_0)$ defined on the sphere S^2, and we wish to compute its spherical harmonic coefficients $c_{l,m}$:

$$f(\theta, \phi) = \sum_{l=0}^{l_{\text{max}}} c_l^T Y_l(\theta, \phi) = \sum_{l=0}^{l_{\text{max}}} \sum_{m=-l}^{l} c_{l,m} Y_{l,m}(\theta, \phi)$$

By orthonormality of the spherical harmonics, and the annihilation property of the Dirac delta:

$$\begin{aligned} c_{l,m} &= \int f(\theta, \phi) Y_{l,m}(\theta, \phi) \sin\theta d\theta d\phi \\ &= \int \delta(\hat{\mathbf{r}} - \hat{\mathbf{r}}_0) Y_{l,m}(\theta, \phi) \sin\theta d\theta d\phi \\ &= Y_{l,m}(\hat{\mathbf{r}}_0) \end{aligned}$$

Thus, we can easily compute the spherical harmonic coefficients for the Dirac delta distribution upto any required l_{max}. This is implemented in the `e3nn-jax` package. Due to the frequency cutoff, the Dirac delta distribution thus obtained is a smooth approximation of a true Dirac delta.

Appendix D

Ablation Studies

D.1 Ablation: l_{max} and Number of Position Channels

To understand the practical effect of adding multiple position channels to Symphony, as well as the impact of increasing l_{max}, we trained variants of Symphony varying l_{max} for the focus embedder E3SchNet from 1 to 2, the number of position channels from 1 to 4, and l_{max} for the position embedder NequIP from 1 to 5.

Due to computational constraints, we trained these models for $1,000,000$ steps each, which is $8\times$ lesser than the original model reported in chapter 5. Thus, the validity numbers are slightly lower overall. However, we believe we can still observe important trends from this experiment.

We report the validity as measured by `xyz2mol` for each of these models in Figure D.1.

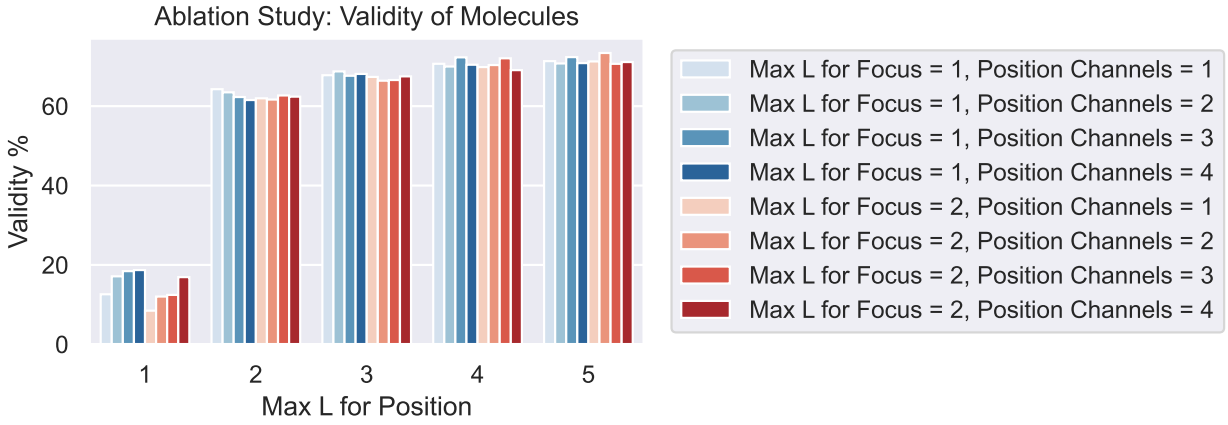

Figure D.1: Validity as a function of l_{max} for the position and focus embedders. Models for which $l_{\text{max}} = 1$ for the focus embedder are marked in blue. Models for which $l_{\text{max}} = 2$ for the focus embedder are marked in red. The intensity of colours increases with the number of position channels.

- For the focus embedder E3SchNet, we do not see a significant increase in validity when going from $l_{\text{max}} = 1$ to $l_{\text{max}} = 2$.

- For the position embedder NequIP, we find a large jump when going from $l_{\text{max}} = 1$ to $l_{\text{max}} = 2$. Further increasing l_{max} seemed to help slightly. For computational reasons, we kept $l_{\text{max}} = 5$.
- Increasing the number of position channels helps for $l_{\text{max}} = 1$ in particular.

D.2 Ablation: Training and Sampling Resolution

Here, we take the trained Symphony model, freeze all weights, and measure the validity of molecules across a range of grid resolutions. The original grid resolution for model training was $(r_\theta, r_\phi) = (180, 359)$ as described above. From Figure D.2, we see that the validity is within the expected variation even when using upto 10× smaller grids. Further amplification of the resolution also does not seem to affect the validity. We hypothesize that this is due to sampling with a lower temperature than ideal making the target distribution more diffuse; future work will seek to understand this effect better.

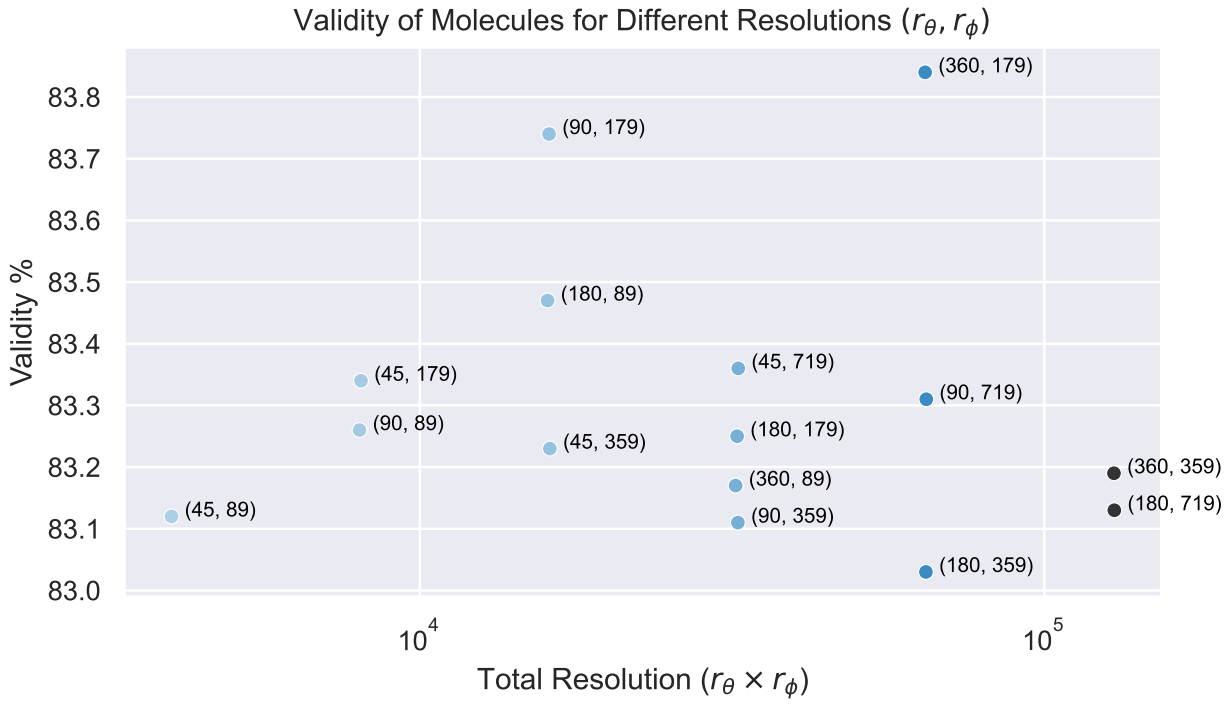

Figure D.2: Validity as a function of sampling grid resolution (r_θ, r_ϕ).

The previous experiment measured the effect of the grid resolution for sampling. We also seeked to understand the effect of the grid resolution for training. For this, we reuse the task of Section B.2, and vary the grid resolution. All other hyperparameters were kept fixed, with $l_{\text{max}} = 2$ and 2 position channels. From Figure D.3, we see that the learning is not affected even at low resolutions. In fact, from a KL divergence perspective, it is easier to learn at lower resolutions because localization is easier. However, lower resolutions come with decreased accuracy when sampling, as shown by the rightmost plot of Figure D.3.

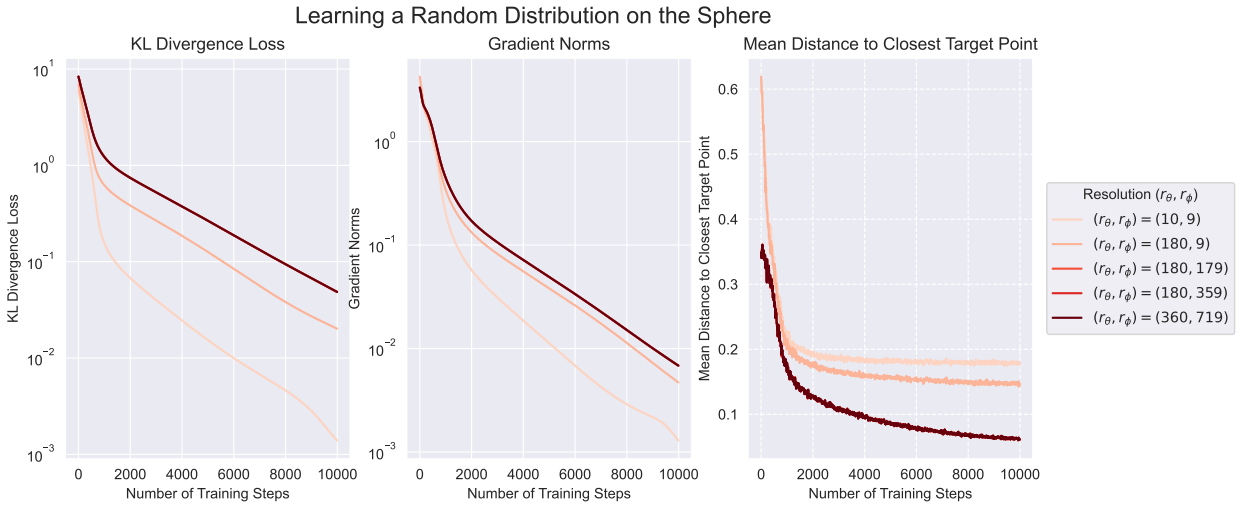

Figure D.3: The effect of resolution when learning the random signal from Section B.2. Our original model was trained with a resolution of $(r_\theta, r_\phi) = (180, 359)$.

D.3 Ablation: Sampling Temperature

Again, we take the trained Symphony model, freeze all weights, and measure the validity of molecules across a range of temperatures T. This means scaling all the logits by a factor of $\frac{1}{T}$. Higher temperatures make the model more diffuse, while lower temperatures make the model more peaked. We see that while the validity improves significantly at lower temperatures, the uniqueness tends to suffer. As seen in Figure D.4, this experiment suggests a more careful sampling of the temperature to better understand a Pareto frontier between validity and uniqueness.

Figure D.4: Validity as a function of temperature applied to the focus (above) and position (below) distribution logits.

Appendix E

Details for Reproducibility

E.1 Details of Models

E.1.1 Embedders

Here, we describe E3SchNet and NequIP [7] which we use to embed the atoms in each fragment into $E(3)$-equivariant features. As shown in Appendix A, we require these models to be $E(3)$-equivariant.

Both of these models are geometric message-passing neural networks, a type of graph neural network [12, 40] that respects the symmetries of 3D structures. In particular, E3SchNet as the EMBEDDER for the focus and atom type prediction, and NequIP as the EMBEDDER for the position prediction. Unlike previous autoregressive models which utilized a shared embedder for all tasks, we found that using different embedders for these two tasks performed much better in our experiments.

Given the fragment $\mathcal{S}^n$, we define the neighbour of each atom $i \in \mathcal{S}^n$ by a Euclidean distance cutoff $\leq d_{\text{max}}$:

$$\mathcal{N}(i) = \{j \in \mathcal{S}^n \mid \|\vec{\mathbf{r}}_{ij}\| \leq d_{\text{max}}\} \tag{E.1}$$

Initially, the features $h_i^{(0)}$ of each atom i in $\mathcal{S}^n$ are set as the embedding of its atomic number Z_i. At each iteration t, the features $h_i^{(t)}$ is updated using the atom's features $h_i^{(t-1)}$ and its neighbour's features $h_j^{(t-1)}$ where $j \in \mathcal{N}(i)$ from the previous round. The final embedding for atom i is returned as $h_i^{(T)}$ where T is the number of message-passing iterations. Algorithm 2 formally shows the operations of a general message passing neural network.

Different message-passing networks differ in their choice of UPDATE function. Following Batzner et al. [7], the UPDATE for NequIP is defined as:

$$\text{UPDATE}(h_i^{(t-1)}, h_{\mathcal{N}(i)}^{(t-1)}) = h_i^{(t-1)} + \frac{1}{C} \sum_{j \in \mathcal{N}(i)} \sum_{l=0}^{l_{\text{max}}} R_\Theta(\|\vec{\mathbf{r}}_{ij}\|) Y^l(\hat{\mathbf{r}}_{ij}) \otimes h_j^{(t-1)}$$

$R_\Theta(\cdot)$ is a learned multi-layer perceptron (MLP). $\otimes$ represents the Clebsch-Gordan Tensor product, which reduces a tensor product into a direct sum of irreducible representations of $O(3)$. We set $C = 20$, $d_{\text{max}} = 5\text{A}$, $l_{\text{max}} = 5$, and $T = 3$ here.

Algorithm 2 General Operation of a Message Passing Neural Network

Input: Fragment $\mathcal{S}^n$, Message Passing Iterations T, Cutoff d_{max}, Update Function UPDATE

Compute neighbor lists for each atom in $\mathcal{S}^n$ according to Equation E.1.

for $i = 1, 2, \ldots, n$ **do**:

$h_i^{(0)} \leftarrow \text{SCALAREMBEDDING}(Z_i)$

for $t = 1, 2, \ldots, T$ **do**:

for $i = 1, 2, \ldots, n$ **do**:

$h_{\mathcal{N}(i)}^{(t-1)} \leftarrow \{h_j^{(t-1)} \mid j \in \mathcal{N}(i)\}$

$h_i^{(t)} \leftarrow \text{UPDATE}(h_i^{(t-1)}, h_{\mathcal{N}(i)}^{(t-1)})$

return $\{h_i^{(T)}\}_{i=1}^n$

E3SchNet is our generalization of the SchNet model [45] that was used in [15] to produce higher-degree $E(3)$-equivariant features. The UPDATE function for E3SchNet is defined as:

$$\text{UPDATE}(h_i^{(t-1)}, h_{\mathcal{N}(i)}^{(t-1)}) = h_i^{(t-1)} + \text{LINEAR}\left(\sum_{j \in \mathcal{N}(i)} \sum_{l=0}^{l_{\text{max}}} W_{ijl} \cdot \left(h_j^{(t-1)} \otimes Y^l(\hat{\mathbf{r}}_{ij})\right)\right)$$

where W_{ijl} are scalars computed via:

$$W_{ijl} = \text{LINEAR}(\sigma(\text{CUTOFF}(\|\vec{\mathbf{r}}_{ij}\|) \cdot \text{RADIALBASIS}(\|\vec{\mathbf{r}}_{ij}\|)))$$

We use the Gaussian radial basis functions, following SchNet. In fact, for $l_{\text{max}} = 0$, E3SchNet reduces exactly to the standard SchNet. We set $l_{\text{max}} = 2$, as we find that the benefits of using even higher degree features for the focus and atom type prediction task are minimal. The cutoff is again 5A.

We see that NequIP and E3SchNet guarantee permutation-equivariance, translation invariance and $O(3)$-equivariance, and hence satisfy the requirements for EMBEDDER in Appendix A.

We implement Symphony with the `e3nn-jax` library that utilizes the JAX [9] framework for creating efficient $E(3)$-equivariant machine learning models.

E.1.2 Training Details

We set $\sigma^2_{\text{true}} = 10^{-5}$ and express the Dirac delta distribution in the spherical harmonic basis upto $l_{\text{max}} = 5$, as explained in Section C.3. The predicted distributions $p^{\text{focus}}, p^{\text{species}}$ and p^{pos} are learned by minimizing the KL divergence to their true counterparts. We found that adding a small amount of zero-centered Gaussian noise $\sigma^2 = 2.5 \times 10^{-3}$ to all input atom positions helped with robustness. All parameters in the EMBEDDER, MLP and LINEAR layers are trained with the Adam [25] optimizer with a learning rate of 5×10^{-4}. We chose the parameters that achieved the lowest loss on the validation set over 8000000 training steps with a batch size of 16 fragments.

E.1.3 Data Details

Following EDM [20], we obtained the QM9 [38] dataset using the DeepChem library [35], and filtered out 3054 'uncharacterized' molecules (available at https://springernature.figshare.com/

ndownloader/files/3195404) which rearranged significantly during geometry optimization, giving us exactly 130831 molecules. Symphony was trained used the same splits as EDM: 100000 molecules to train, 13083 molecules for validation and 17748 molecules for test, obtained from a random permutation of the molecules.

E.1.4 Baseline Model Details

For the baseline models, we used the pretrained EDM model at https://github.com/ehoogeboom/e3_diffusion_for_molecules and the pretrained G-SphereNet model at https://github.com/divelab/DIG/tree/dig-stable/examples/ggraph3D/G_SphereNet. We retrained the G-SchNet model on the EDM splits following https://github.com/atomistic-machine-learning/G-SchNet. The samples (in `.xyz` format) of all models used for evaluation is available at this URL: https://figshare.com/s/a17ccface17f0c22f15a.

Our JAX code containing all of the data preprocessing, model training and evaluation metrics is available at https://github.com/atomicarchitects/symphony.

E.2 Details of Metrics

E.2.1 PoseBusters

Table E.1 provides details of the Posebusters tests used in Table 5.2. We use the default parameters from their framework.

Test	Description
All Atoms Connected	There exists a path along bonds between any two atoms in the molecule.
Reasonable Bond Lengths	The bond lengths in the input molecule are within 0.75 of the lower and 1.25 of the upper bounds determined by distance geometry.
Reasonable Bond Angles	The angles in the input molecule are within 0.75 of the lower and 1.25 of the upper bounds determined by distance geometry.
Aromatic Rings Flatness	All atoms in aromatic rings with 5 or 6 members are within 0.25A of the closest shared plane.
Double Bonds Flatness	The two carbons of aliphatic carbon-carbon double bonds and their four neighbours are within 0.25A of the closest shared plane.
Reasonable Molecule Energy	The calculated energy of the input molecule is no more than 100 times the average energy of an ensemble of 50 conformations generated for the input molecule. The energy is calculated using the UFF [36] in RDKit [27] and the conformations are generated with ETKDGv3 [37] followed by force field relaxation using the UFF with up to 200 iterations.
No Internal Steric Clash	The interatomic distance between pairs of non-covalently bound atoms is above 0.8 of the lower bound determined by distance geometry.

Table E.1: Description of each intramolecular PoseBusters test, taken from Table 4 of Buttenschoen et al. [10].

E.2.2 Maximum Mean Discrepancy

The Maximum Mean Discrepancy (MMD), introduced in Gretton et al. [18], measures how different two distributions p_X and p_Y are, given a kernel function k. Formally, the MMD is defined as:

$$\mathrm{MMD}(p_X, p_Y) = \sqrt{\mathop{\mathbb{E}}_{X,X'\sim p_X}[k(X,X')] + \mathop{\mathbb{E}}_{Y,Y'\sim p_X}[k(Y,Y')] - \mathop{\mathbb{E}}_{X\sim p_X, Y\sim p_Y}[k(X,Y)]}$$

From the above equation, we see that the MMD can be easily estimated with samples from each distribution. We choose k as the sum of Gaussian kernels at different scales:

$$k(X, X') = \sum_{i=0}^{29} \exp(-10^{\left(\frac{i}{5}-3\right)} \cdot \|X - X'\|^2)$$

Bibliography

[1] Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions, 2023.

[2] Brandon Anderson, Truong-Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks, 2019.

[3] Brian. D. O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12:313–326, 1982. URL https://api.semanticscholar.org/CorpusID:3897405.

[4] Dylan M. Anstine and Olexandr Isayev. Generative Models as an Emerging Paradigm in the Chemical Sciences. *Journal of the American Chemical Society*, 145(16):8736–8750, 2023. doi:10.1021/jacs.2c13467. URL https://doi.org/10.1021/jacs.2c13467. PMID: 37052978.

[5] Michael Banck, Craig A. Morley, Tim Vandermeersch, and Geoffrey R. Hutchison. Open Babel: An open chemical toolbox, 2011. URL https://doi.org/10.1186/1758-2946-3-33.

[6] Ilyes Batatia, David Peter Kovacs, Gregor N. C. Simm, Christoph Ortner, and Gabor Csanyi. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=YPpSngE-ZU.

[7] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. 13, May 2022. URL https://doi.org/10.1038/s41467-022-29939-5.

[8] Erik J Bekkers, Sharvaree Vadgama, Rob Hesselink, Putri A Van der Linden, and David W. Romero. Fast, expressive se(n) equivariant networks through weight-sharing in position-orientation space. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dPHLbUqGbr.

[9] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable Transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[10] Martin Buttenschoen, Garrett M. Morris, and Charlotte M. Deane. PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences, 2023.

[11] Taco S. Cohen and Max Welling. Harmonic exponential families on manifolds, 2015.

[12] Ameya Daigavane, Balaraman Ravindran, and Gaurav Aggarwal. Understanding Convolutions on Graphs. *Distill*, 2021. doi:10.23915/distill.00032. https://distill.pub/2021/understanding-gnns.

[13] Ameya Daigavane, Song Eun Kim, Mario Geiger, and Tess Smidt. Symphony: Symmetry-equivariant point-centered spherical harmonics for 3d molecule generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MIEnYtlGyv.

[14] Octavian-Eugen Ganea, Lagnajit Pattanaik, Connor W. Coley, Regina Barzilay, Klavs F. Jensen, William H. Green, and Tommi S. Jaakkola. Geomol: Torsional geometric generation of molecular 3d conformer ensembles, 2021.

[15] Niklas Gebauer, Michael Gastegger, and Kristof Schütt. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/a4d8e2a7e0d0c102339f97716d2fdfb6-Paper.pdf.

[16] Niklas W. A. Gebauer, Michael Gastegger, Stefaan S. P. Hessmann, Klaus-Robert Müller, and Kristof T. Schütt. Inverse design of 3d molecular structures with conditional generative neural networks. 13, February 2022. URL https://doi.org/10.1038/s41467-022-28526-y.

[17] Mario Geiger and Tess Smidt. e3nn: Euclidean Neural Networks, 2022.

[18] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL http://jmlr.org/papers/v13/gretton12a.html.

[19] Jiaqi Guan, Wesley Wei Qian, qiang liu, Wei-Ying Ma, Jianzhu Ma, and Jian Peng. Energy-inspired molecular conformation optimization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=7QfLW-XZTl.

[20] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant Diffusion for Molecule Generation in 3D, 2022.

[21] John B. Ingraham, Max Baranov, Zak Costello, Karl W. Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M. Lord, Christopher Ng-Thow-Hing, Erik R. Van Vlack, Shan Tie, Vincent Xue, Sarah C. Cowles, Alan Leung, João V. Rodrigues, Claudio L. Morales-Perez, Alex M. Ayoub, Robin Green, Katherine Puentes, Frank Oplinger, Nishant V. Panwar, Fritz Obermeyer, Adam R. Root, Andrew L. Beam, Frank J. Poelwijk, and Gevorg Grigoryan. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

[22] Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation, 2023.

[23] Bowen Jing, Bonnie Berger, and Tommi Jaakkola. Alphafold meets flow matching for generating protein ensembles, 2024.

[24] Yeonjoon Kim and Woo Youn Kim. Universal Structure Conversion Method for Organic Molecules: From Atomic Connectivity to Three-Dimensional Geometry. *Bulletin of the Korean Chemical Society*, 36(7):1769–1777, 2015. doi:https://doi.org/10.1002/bkcs.10334. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/bkcs.10334.

[25] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2017.

[26] Romain Lacombe and Neal Vaidya. Accelerating the generation of molecular conformations with progressive distillation of equivariant latent diffusion models, 2024.

[27] Greg Landrum, Paolo Tosco, Brian Kelley, Ric, David Cosgrove, sriniker, gedeck, Riccardo Vianello, Nadine Schneider, Eisuke Kawashima, Dan N, Gareth Jones, Andrew Dalke, Brian Cole, Matt Swain, Samo Turk, Alexander Savelyev, Alain Vaucher, Maciej Wójcikowski, Ichiru Take, Daniel Probst, Kazuya Ujihara, Vincent F. Scalfani, guillaume godin, Juuso Lehtivarjo, Axel Pahl, Rachel Walker, Francois Berenger, jasondbiggs, and strets123. rdkit/rdkit: 2023_03_2 (Q1 2023) Release, June 2023. URL https://doi.org/10.5281/zenodo.8053810.

[28] Tuan Le, Julian Cremer, Frank Noé, Djork-Arné Clevert, and Kristof Schütt. Navigating the design space of equivariant diffusion-based generative models for de novo 3d molecule generation, 2023.

[29] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023.

[30] Youzhi Luo and Shuiwang Ji. An Autoregressive Flow Model for 3D Molecular Geometry Generation from Scratch. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=C03Ajc-NS5W.

[31] Alex Morehead and Jianlin Cheng. Geometry-complete diffusion for 3d molecule generation and optimization, 2024.

[32] Kieran Murphy, Carlos Esteves, Varun Jampani, Srikumar Ramalingam, and Ameesh Makadia. Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold, 2022.

[33] Albert Musaelian, Simon Batzner, Anders Johansson, Lixin Sun, Cameron J. Owen, Mordechai Kornbluth, and Boris Kozinsky. Learning local equivariant representations for large-scale atomistic dynamics. *Nature Communications*, 14(1):579, 2023.

[34] Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2mol: Efficient molecular sampling based on 3d protein pockets, 2022.

[35] Bharath Ramsundar, Peter Eastman, Patrick Walters, Vijay Pande, Karl Leswing, and Zhenqin Wu. *Deep Learning for the Life Sciences*. O'Reilly Media, 2019. https://www.amazon.com/Deep-Learning-Life-Sciences-Microscopy/dp/1492039837.

[36] A. K. Rappe, C. J. Casewit, K. S. Colwell, W. A. III Goddard, and W. M. Skiff. UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American Chemical Society*, 114(25):10024–10035, 12 1992. doi:10.1021/ja00051a040. URL https://doi.org/10.1021/ja00051a040.

[37] Sereina Riniker and Gregory A. Landrum. Better Informed Distance Geometry: Using What We Know To Improve Conformation Generation. *Journal of Chemical Information and Modeling*, 55(12):2562–2574, 2015. doi:10.1021/acs.jcim.5b00654. URL https://doi.org/10.1021/acs.jcim.5b00654. PMID: 26575315.

[38] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108: 058301, 2012.

[39] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models, 2022.

[40] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B. Wiltschko. A Gentle Introduction to Graph Neural Networks. *Distill*, 2021. doi:10.23915/distill.00033. https://distill.pub/2021/gnn-intro.

[41] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) Equivariant Graph Neural Networks, 2022.

[42] Arne Schneuing, Yuanqi Du, Charles Harris, Arian Jamasb, Ilia Igashov, Weitao Du, Tom Blundell, Pietro Lió, Carla Gomes, Max Welling, Michael Bronstein, and Bruno Correia. Structure-based drug design with equivariant diffusion models, 2023.

[43] Schrödinger, LLC. The PyMOL Molecular Graphics System, Version 1.8. November 2015.

[44] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[45] Kristof T. Schütt, Pieter-Jan Kindermans, Huziel E. Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions, 2017.

[46] Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation, 2021.

[47] Gregor Simm, Robert Pinsler, and Jose Miguel Hernandez-Lobato. Reinforcement learning for molecular design guided by quantum mechanics. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8959–8969. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/simm20b.html.

[48] Gregor N. C. Simm and José Miguel Hernández-Lobato. A generative model for molecular distance geometry, 2020.

[49] Gregor N. C. Simm, Robert Pinsler, Gábor Csányi, and José Miguel Hernández-Lobato. Symmetry-aware actor-critic for 3d molecular design. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=jEYKjPE1xYN.

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

[51] Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant flow matching with hybrid probability transport for 3d molecule generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=hHUZ5V9XFu.

[52] Martin Uhrin. Through the eyes of a descriptor: Constructing complete, invertible descriptions of atomic environments. *Phys. Rev. B*, 104:144110, Oct 2021. doi:10.1103/PhysRevB.104.144110. URL https://link.aps.org/doi/10.1103/PhysRevB.104.144110.

[53] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete Denoising diffusion for graph generation, 2023.

[54] Clement Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. MiDi: Mixed Graph and 3D Denoising Diffusion for Molecule Generation, 2023.

[55] Yuyang Wang, Ahmed A. Elhag, Navdeep Jaitly, Joshua M. Susskind, and Miguel Angel Bautista. Generating molecular conformer fields, 2023.

[56] David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 02 1988. doi:10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

[57] Minkai Xu, Shitong Luo, Yoshua Bengio, Jian Peng, and Jian Tang. Learning neural generative dynamics for molecular conformation generation, 2021.

[58] Minkai Xu, Wujie Wang, Shitong Luo, Chence Shi, Yoshua Bengio, Rafael Gomez-Bombarelli, and Jian Tang. An end-to-end framework for molecular conformation generation via bilevel programming, 2021.

[59] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. GeoDiff: a Geometric Diffusion Model for Molecular Conformation Generation, 2022.