SE(3) Equivariant Augmented Coupling Flows

Laurence I. Midgley
University of Cambridge
lim24@cam.ac.uk

Vincent Stimper
Max Planck Institute for Intelligent Systems
University of Cambridge
vs488@cam.ac.uk

Javier Antorán
University of Cambridge
ja666@cam.ac.uk

Emile Mathieu
University of Cambridge
ebm32@cam.ac.uk

Bernhard Schölkopf
Max Planck Institute
for Intelligent Systems
bs@tue.mpg.de

José Miguel Hernández-Lobato
University of Cambridge
jmh233@cam.ac.uk

Abstract

Coupling normalizing flows allow for fast sampling and density evaluation, making them the tool of choice for probabilistic modeling of physical systems. However, the standard coupling architecture precludes endowing flows that operate on the Cartesian coordinates of atoms with the SE(3) and permutation invariances of physical systems. This work proposes a coupling flow that preserves SE(3) and permutation equivariance by performing coordinate splits along additional augmented dimensions. At each layer, the flow maps atoms' positions into learned SE(3) invariant bases, where we apply standard flow transformations, such as monotonic rational-quadratic splines, before returning to the original basis. Crucially, our flow preserves fast sampling and density evaluation, and may be used to produce unbiased estimates of expectations with respect to the target distribution via importance sampling. When trained on the DW4, LJ13 and QM9-positional datasets, our flow is competitive with equivariant continuous normalizing flows, while allowing sampling two orders of magnitude faster. Moreover, to the best of our knowledge, we are the first to learn the full Boltzmann distribution of alanine dipeptide by only modeling the Cartesian positions of its atoms. Lastly, we demonstrate that our flow can be trained to approximately sample from the Boltzmann distribution of the DW4 and LJ13 particle systems using only their energy functions.

1 Introduction

Modeling the distribution of a molecule's configurations at equilibrium, known as the Boltzmann distribution, is a promising application of deep generative models [Noé et al., 2019]. While the unnormalized density of the Boltzmann distribution can be obtained via physical modeling, sampling from it typically requires molecular dynamics (MD) simulations, which are expensive and produce correlated samples. A promising alternative is to rely on surrogate deep generative models, known as a Boltzmann generators. We can draw independent samples from these and debias any expectations estimated with the samples via importance weighting. The Boltzmann distribution typically admits rotation and translation symmetries, known as the Special Euclidean group $\mathrm{SE}(3)$, as well as permutation symmetry. These constraints are important to incorporate into the model as they improve

Preprint. Under review.

training efficiency and generalization [Cohen and Welling, 2016, Batzner et al., 2022a, Köhler et al., 2020]. Other key desiderata of Boltzmann generators are that they allow for fast sampling and density evaluation. These are necessary for energy-based training, that is, training using the Boltzmann distribution's unnormalized density [Noé et al., 2019, Stimper et al., 2022, Midgley et al., 2023]. Training by energy is critical, as it prevents the model quality from being constrained by the quality and quantity of MD samples.

Existing coupling flows which approximate the Boltzmann distribution of molecules are at least partially defined over internal coordinates, i.e. bond distances, bond angles, and dihedral angles [Wu et al., 2020, Campbell et al., 2021, Köhler et al., 2023a, Midgley et al., 2023], which are $\mathrm{SE}(3)$ invariant. However, the definition of these depends on the molecular graph and they are non-unique for most graphs. Furthermore, models parameterized in internal coordinates struggle to capture interactions among nodes far apart in the graph and cannot capture the permutation invariance of some atoms. Thus they are not suitable for particle systems such as LJ13 [Köhler et al., 2020]. $\mathrm{SE}(3)$ equivariance constraints have been applied to continuous normalizing flows (CNFs) operating on Cartesian coordinates [Köhler et al., 2020, Satorras et al., 2021a], and to the closely related diffusion models [Xu et al., 2022, Hoogeboom et al., 2022, Yim et al., 2023]. These models are built upon $\mathrm{SE}(3)$ equivariant graph neural networks (GNNs) [Satorras et al., 2021b, Geiger and Smidt, 2022, Batzner et al., 2022b]. These architectures can be applied to any molecular graph [Jing et al., 2022], enabling a single generative model to generalize across many molecules. Alas, sampling and evaluating the density of CNFs and diffusion models typically requires thousands of neural network evaluations [Xiao et al., 2022], preventing them from being trained by energy. As such, presently no Boltzmann generator exists that (i) acts on euclidean coordinates of atoms (ii) enforces $\mathrm{SE}(3)$ equivariance, and (iii) allows for fast sampling.

To address this gap, we propose a flexible $\mathrm{SE}(3)$ equivariant coupling flow that operates on the Cartesian coordinates of atoms, allowing for fast sampling and density evaluation. Our contributions are:

- We extend coupling layers to be $\mathrm{SE}(3)$ equivariant by augmenting their input space with auxiliary variables [Huang et al., 2020] which can be acted upon on by $\mathrm{SE}(3)$. We update the atom positions conditioned on the auxiliary variables by first projecting the atoms into an $\mathrm{SE}(3)$-invariant space and then applying a standard normalizing flow transform before projecting its output back onto the equivariant space.
- We demonstrate that, when trained by maximum likelihood, our flow matches the performance of both existing $\mathrm{SE}(3)$ CNFs and coupling flows operating on internal coordinates on molecular generation tasks. Our flow is more than 100 times faster to sample from than $\mathrm{SE}(3)$ CNFs. We are the first to learn the full Boltzmann distribution of alanine dipeptide solely in Cartesian coordinates.
- We demonstrate our flow in the energy-based training setting on the DW4 and LJ13 problems, where parameters are learned using only the molecular energy function. Energy-based training of the CNF is intractable due to slow sampling and density evaluation. Flows that operate on internal coordinates are not able to capture the permutation invariance of these problems. Hence, our flow is the only existing permutation and $\mathrm{SE}(3)$ equivariant method that can tractably be applied there.

2 Background: coupling flows and invariant models

2.1 Normalizing flows and coupling transforms

A (discrete-time) normalizing flow is a flexible parametric family of densities on $\mathcal{X}$ as the push-forward of a base density q_0 along an invertible automorphism $f_\theta : \mathcal{X} \to \mathcal{X}$ with parameters $\theta \in \Theta$ [Papamakarios et al., 2021]. The density is given by the change of variable formula:

$$q_\theta(x) = q_0(f^{-1}(x)) \left| \det \frac{\partial f_\theta^{-1}(x)}{\partial x} \right|. \tag{1}$$

We can efficiently sample from the flow by sampling from q_0 and mapping these samples through f_θ in a single forward pass. A popular way to construct f_θ is to use coupling transforms. The D dimensional input $\boldsymbol{x} \in \mathcal{X}$ is split into two sets, transforming the first set conditional on the second, while leaving the second set unchanged:

$$\begin{aligned} \boldsymbol{y}_{1:d} &= \mathcal{T}(\boldsymbol{x}_{1:d}; \boldsymbol{x}_{d+1:D}), \\ \boldsymbol{y}_{d+1:D} &= \boldsymbol{x}_{d+1:D}. \end{aligned} \tag{2}$$

They induce a lower triangular Jacobian, such that its determinant becomes $|\partial \mathcal{T}(\boldsymbol{x}_{1:d}; \boldsymbol{x}_{d+1:D})/\partial \boldsymbol{x}_{1:d}|$. Further, choosing $\mathcal{T}$ to have an easy to compute determinant, such as an elementwise transformation [Dinh et al., 2015, 2017, Durkan et al., 2019], allows for fast density evaluation and sampling at low computational cost.

2.2 Equivariance and invariance for coupling flow models of molecular conformations

Throughout, we deal with observations of an n-body system represented by a matrix $\boldsymbol{x} = \left[x^1, \ldots, x^n\right] \in \mathcal{X} = \mathbb{R}^{3\times n}$, where the rows index Cartesian coordinates and the columns index individual particles. We seek to construct flows on $\mathcal{X}$ endowed with the symmetries present in molecular energy functions. These are invariant to rotations and translations of $\boldsymbol{x}$ ($\mathrm{SE}(3)$), and to permutation of atoms of the same type (S_n). We will formalize them in this section.

Symmetry groups The special Euclidean group $\mathrm{SE}(3)$ is the set of orientation preserving rigid transformations in Euclidean space. Its elements $t \in \mathrm{SE}(3)$ can be decomposed into two components $t = (R, u)$ where $R \in \mathrm{SO}(3)$ is a 3×3 rotation matrix and $u \in \mathbb{R}^3$ represents a translation; for a coordinate $v \in \mathbb{R}^3$, $t \cdot v = Rv + u$ denotes the action of t on v. The symmetric group S_n defined over a set of n atoms consists of all $n!$ permutations that can be performed with said atoms. Its elements $\sigma \in S_n$ act on an n-body system as $\sigma \cdot \boldsymbol{x} = [x^{\sigma(1)}, \ldots, x^{\sigma(n)}]$.

Equivariant maps A map $f : \mathcal{X} \to \mathcal{Y}$ is said to be *G-equivariant* if it commutes with the group action $\cdot$, i.e. if for any $x \in \mathcal{X}$ and $g \in G$ we have $f(g \cdot x) = g \cdot f(x)$. Invariance is a special case where for any $x \in \mathcal{X}$ and $g \in G$ we have $f(g \cdot x) = f(x)$. There has been a plethora of recent work on constructing graph neural network functions equivariant to the action of $G = \mathrm{SE}(3) \times S_n$ [e.g. Thomas et al., 2018, Satorras et al., 2021b, Geiger and Smidt, 2022], which we will leverage to construct our equivariant coupling flow model.

Invariant density A density $p : \mathcal{X} \to \mathbb{R}_+$ is G-invariant if for any $x \in \mathcal{X}$ and $g \in G$ we have $p(g \cdot x) = p(x)$. Combining an invariant base density q_0 with an equivariant invertible transform f, as in (1), yields an invariant flow density [Papamakarios et al., 2021, Köhler et al., 2020]. This gives a practical way to design invariant densities models.

Challenges in constructing $\mathrm{SO}(3) \times S_n$ invariant flow models Unfortunately, no coupling transform can be simultaneously equivariant to both permutation of the particles and their rotations; coupling splits must be performed either across particles or spatial dimensions which would break either permutation or rotational symmetry [Köhler et al., 2020, Bose et al., 2022].

Furthermore, there does not exist a translation invariant *probability* measure, as any such measure would be proportional to the Lebesgue measure and therefore not have unit volume. This precludes us from defining an invariant base distribution directly on $\mathcal{X}$. Fortunately, Proposition A.1 and its converse allow us to disintegrate the probability measure into a translational measure proportional to the Lebesgue measure and an $\mathrm{SO}(3)$-invariant probability measure on the subspace of $\mathbb{R}^{3\times n}$ with zero center of mass. We can drop the former and only model the latter.

3 Method: $\mathrm{SE}(3) \times S_n$ equivariant augmented coupling flow model

This section describes our main contribution, an $\mathrm{SE}(3) \times S_n$ equivariant coupling flow. We first lay the groundwork for achieving translation invariance by defining our flow density on a lower-dimensional "zero Center of Mass (CoM)" space. To preserve permutation and rotation equivariance, we leverage the augmented flow framework of Huang et al. [2020]. Specifically, we use sets of augmented variables as a pivot for coupling transforms. Sec. 3.1 introduces a novel class of coupling transforms that achieve the aforementioned permutation and rotation equivariance by operating on atoms projected using a set of equivariant bases. Sec. 3.2 describes our choice of invariant base distribution and, finally, in Sec. 3.3, we discuss several schemes to train the augmented flow from either samples or energy functions and how to perform efficient density evaluation.

Translation invariance is obtained by modelling the data on the quotient space $\mathbb{R}^{3\times n}/\,\mathbb{R}^3 \triangleq \tilde{\mathcal{X}} \subseteq \mathcal{X}$, where all n-body systems that only differ by a translation are "glued" together, i.e. where $\boldsymbol{x} \sim \boldsymbol{x}'$ if $\boldsymbol{x} = \boldsymbol{x}' + p$ with $p \in \mathbb{R}^3$. Constructing a parametric probabilistic model over $\tilde{\mathcal{X}}$, automatically endowing it with translation invariance. In practice, we still work with Cartesian coordinates, but

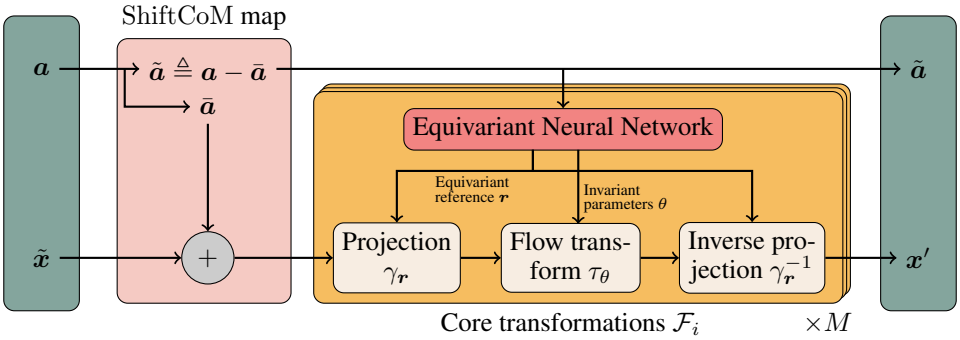


Figure 1: Illustration of the equivariant coupling layer of our augmented normalizing flow, where our variable with zero center of mass (CoM) $\tilde{\boldsymbol{x}}$ is transformed with the augmented variable $\boldsymbol{a}$.

center the data so as to zero its CoM: $\tilde{\boldsymbol{x}} \triangleq \boldsymbol{x} - \bar{\boldsymbol{x}}$ with $\bar{\boldsymbol{x}} \triangleq \frac{1}{n}\sum_{i=1}^{n}[\boldsymbol{x}]^i$. Thus, $\tilde{\boldsymbol{x}}$ lies on $\tilde{\mathcal{X}}$, an $(n-1)\times 3$ dimensional hyperplane embedded in $\mathcal{X}$.

Augmented variable pivoted coupling To allow for simultaneously permutation and rotation equivariant coupling transforms, we introduce *augmented* variables $\boldsymbol{a} \in \mathcal{A}$. Our coupling layers update the particle positions $\boldsymbol{x}$ conditioned on $\boldsymbol{a}$ and vice-versa. The augmented variables need to "behave" similarly to $\boldsymbol{x}$, in that they can also be acted upon by elements of $\mathrm{SE}(3) \times S_n$. We achieve this by choosing $\boldsymbol{a}$ to be a set of k of observation-sized arrays $\mathcal{A} = \mathcal{X}^k$, which we will discuss further in App. B.4. Importantly, we do not restrict $\boldsymbol{a}$ to be zero-CoM.

Invariant flow density on the extended space We parameterize a density q over the extended space $\tilde{\mathcal{X}} \times \mathcal{A}$ w.r.t. the product measure $\lambda_{\tilde{\mathcal{X}}} \otimes \lambda_{\mathcal{A}}$, where $\lambda_{\tilde{\mathcal{X}}} \in \mathcal{P}(\tilde{\mathcal{X}})$ and $\lambda_{\mathcal{A}} \in \mathcal{P}(\mathcal{A})$ respectively denote the Lebesgue measure on $\tilde{\mathcal{X}}$ and $\mathcal{A}$. We use $q(\boldsymbol{x}, \boldsymbol{a})$ as a shorthand for the density of the corresponding zero-CoM projection $q(\tilde{\boldsymbol{x}}, \boldsymbol{a})$. The density q is constructed to be invariant to the action of $G \triangleq \mathrm{SO}(3) \times S_n$ when simultaneously applied to the observed and augmented variables, that is, $q(\boldsymbol{x}, \boldsymbol{a}) = q(g \cdot \boldsymbol{x}, g \cdot \boldsymbol{a})$ for any $g \in G$. We aim to construct a G-equivariant flow $f : \tilde{\mathcal{X}} \times \mathcal{A} \to \tilde{\mathcal{X}} \times \mathcal{A}$ on this extended space, and combine it with $q_0 : \tilde{\mathcal{X}} \times \mathcal{A} \to \mathbb{R}^+$, a G-invariant base density function, to yield the invariant flow density

$$q(\boldsymbol{x}, \boldsymbol{a}) = q_0(f^{-1}(\boldsymbol{x}, \boldsymbol{a})) \left| \det \frac{\partial f^{-1}(\boldsymbol{x}, \boldsymbol{a})}{\partial(\boldsymbol{x}, \boldsymbol{a}))} \right|. \tag{3}$$

Proposition 3.1 (Invariant marginal)**.** *Assume $q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}_+$ is a G-invariant density over the probability space $(\mathcal{X} \times \mathcal{A}, \lambda_{\mathcal{X}} \otimes \lambda_{\mathcal{A}})$, then $q_{\boldsymbol{x}} \triangleq \int_{\mathcal{A}} q(\cdot, \boldsymbol{a}) \lambda_{\mathcal{A}}(\mathrm{d}\boldsymbol{a}) : \mathcal{X} \to \mathbb{R}_+$ is a G-invariant density w.r.t. to the measure $\lambda_{\mathcal{X}}$.*

Proof. For any $g \in G$, $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{a} \in \mathcal{A}$

$$q_{\boldsymbol{x}}(g \cdot \boldsymbol{x}) = \int_{\mathcal{A}} q(g \cdot \boldsymbol{x}, \boldsymbol{a}) \lambda_{\mathcal{A}}(\mathrm{d}\boldsymbol{a}) = \int_{g^{-1}\mathcal{A}} q(g \cdot \boldsymbol{x}, g \cdot \boldsymbol{a}) \lambda_{\mathcal{A}}(\mathrm{d}\, g \cdot \boldsymbol{a}) = \int_{\mathcal{A}} q(\boldsymbol{x}, \boldsymbol{a}) \lambda_{\mathcal{A}}(\mathrm{d}\boldsymbol{a}) = q_{\boldsymbol{x}}(\boldsymbol{x}),$$

where we used the G-invariance of q and of the measure $\lambda_{\mathcal{A}}$, as well as $g^{-1}\mathcal{A} = \mathcal{A}$. □

3.1 SE(3) and permutation equivariant coupling transform

We now derive our $\mathrm{SE}(3) \times S_n$ equivariant map $f : \tilde{\mathcal{X}} \times \mathcal{A} \to \tilde{\mathcal{X}} \times \mathcal{A}$ defined on the extended space. We introduce two modules, a CoM-shift transform, which swaps the center of mass between these observed and augmented variables, and an equivariant core transformation, which updates on $\tilde{\boldsymbol{x}}$ conditional on $\boldsymbol{a}$ and vice versa. Composing them yields our equivariant coupling layer which is illustrated in Fig. 1.

$\mathrm{SE}(3) \times S_n$ equivariant coupling We dissociate the equivariance constraint from the tractability of the Jacobian determinant by constructing a core coupling transformation that composes (a) a parameterized projection onto invariant features γ, (b) a standard normalizing flow transform τ, (c) a projection back into the original space γ^{-1}. Denoting the inputs with superscript ℓ and with $\ell + 1$ for

outputs, our core transformation $\mathcal{F} : (\boldsymbol{x}^\ell; \boldsymbol{a}^\ell) \mapsto (\boldsymbol{x}^{\ell+1}, \boldsymbol{a}^{\ell+1})$ is given as

$$\begin{aligned} \boldsymbol{x}^{\ell+1} &= \gamma_{\boldsymbol{r}}^{-1} \cdot \tau_\theta(\gamma_{\boldsymbol{r}} \cdot \boldsymbol{x}^\ell), \quad \text{with } (\boldsymbol{r}, \theta) = h(\boldsymbol{a}^\ell), \\ \boldsymbol{a}^{\ell+1} &= \boldsymbol{a}^\ell. \end{aligned} \tag{4}$$

Here, h is a (graph) neural network that returns a set of equivariant reference vectors $\boldsymbol{r}$, and invariant parameters θ. The equivariant vectors $\boldsymbol{r}$ parametrize $\gamma_{\boldsymbol{r}}$, a projection operator onto a rotation invariant feature space. This means that any rotations applied to the inputs will be cancelled by $\gamma_{\boldsymbol{r}}$, i.e. $\gamma_{g\cdot\boldsymbol{r}} = \gamma_{\boldsymbol{r}}^{-1} \cdot g^{-1}$ or equivalently $(\gamma_{g\cdot\boldsymbol{r}})^{-1} = g \cdot \gamma_{\boldsymbol{r}}^{-1}$ for all $g \in \mathrm{SO}(3)$. We use the inverse projection $\gamma_{\boldsymbol{r}}^{-1}$ to map the invariant features back to equivariant features. The function τ_θ is a standard flow transformation, that we apply to the invariant features. Its parameters θ are rotation invariant.

We now give sufficient conditions under which our core coupling transform is equivariant.

Proposition 3.2 (Equivariant augmented coupling flow)**.** *If $h : \mathcal{A} \to \mathcal{X}^n \times \Theta^n$ is $\mathrm{SO}(3)$-equivariant for its first output, $\mathrm{SO}(3)$-invariant for its second, and S_n equivariant for both, and $(\gamma_{g\cdot\boldsymbol{r}})^{-1} = g \cdot \gamma_{\boldsymbol{r}}^{-1}$ for all $g \in \mathrm{SO}(3)$, the transform $\mathcal{F}$ given by (7) is $\mathrm{SO}(3) \times S_n$ equivariant.*

Proof. For $\mathrm{SO}(3)$: We first notice that $h(g \cdot \boldsymbol{a}) = (g \cdot \boldsymbol{r}, \theta)$, and then since $(\gamma_{g\cdot\boldsymbol{r}})^{-1} = g \cdot \gamma_{\boldsymbol{r}}^{-1}$ we have $\mathcal{F}(g \cdot \boldsymbol{x}, g \cdot \boldsymbol{a}) = (\gamma_{g\cdot\boldsymbol{r}})^{-1} \cdot \tau_\theta(\gamma_{g\cdot\boldsymbol{r}} \cdot g \cdot \boldsymbol{x}^\ell) = g \cdot \gamma_{\boldsymbol{r}}^{-1} \cdot \tau_\theta(\gamma_{\boldsymbol{r}} \cdot g^{-1} \cdot g \cdot \boldsymbol{x}^\ell) = g \cdot \mathcal{F}(\boldsymbol{x}, \boldsymbol{a})$.

For S_n: We first note that $h(\sigma \cdot \boldsymbol{a}) = (\sigma \cdot \boldsymbol{r}, \sigma \cdot \theta)$. Then, using that $\gamma_{\boldsymbol{r}}$ and τ act on $\boldsymbol{x}$ atom-wise, we have $\mathcal{F}(\sigma\cdot\boldsymbol{x}, \sigma\cdot\boldsymbol{a}) = \gamma_{\sigma\cdot\boldsymbol{r}}^{-1} \cdot \tau_{\sigma\cdot\theta}(\gamma_{\sigma\cdot\boldsymbol{r}} \cdot (\sigma\cdot\boldsymbol{x})) = (\sigma\cdot\gamma_{\boldsymbol{r}}^{-1}) \cdot (\sigma\cdot\tau_\theta)((\sigma\cdot\gamma_{\boldsymbol{r}}) \cdot (\sigma\cdot\boldsymbol{x})) = \sigma \cdot \mathcal{F}(\boldsymbol{x}, \boldsymbol{a})$. □

For the Jacobian of the coupling described above to be well-defined, the variable being transformed must be non-zero CoM (see App. B.1 for a derivation). Thus, although our observations live on $\tilde{\mathcal{X}}$, for now, assume that the inputs to the transform are not zero-CoM and we will deal with this assumption in the following paragraphs. This choice also allows us to use standard equivariant GNNs for h [Satorras et al., 2021b, Geiger and Smidt, 2022] which leverage per-node features defined in the ambient space, such as atom type and molecular graph connectivity.

Choices of projection $\boldsymbol{\gamma}$ The equivariant vectors $\boldsymbol{r}$ parameterize a local (per-atom) $\mathrm{SO}(3)$ equivariant reference frame used in the projection $\gamma_{\boldsymbol{r}}$. We introduce three different projection strategies. (i) The first strategy, is for $\boldsymbol{r}$ to consist of an origin and orthonormal rotation matrix both of which we apply to each atom's positions. We then take τ to be a dimension-wise transformation for each of the projected atoms' coordinates. We dub this method CARTESIAN-PROJ. (ii) Alternatively, we let $\boldsymbol{r}$ parameterize a reference plane for spherical coordinates, as in Liu et al. [2022]. We then apply elementwise transforms to each atom's radius, polar angle and azimuthal angle. We call this SPHERICAL-PROJ. (iii) Lastly, consider a variant of SPHERICAL-PROJ where just the radius is transformed and the polar and azimuth angles are held constant. Here $\boldsymbol{r}$ parameterizes a single reference point, a per-atom origin. We refer to this last variant as VECTOR-PROJ.

Architectural details For the transformations applied in the invariant projected space we consider affine mappings [Dinh et al., 2017] and monotonic rational-quadratic splines [Durkan et al., 2019]. Additionally, to limit computational cost, we have our GNNs h output M sets of reference vectors $\boldsymbol{r}$ and invariant parameters θ. These parametrize M core coupling transformations with a single GNN forward pass. We provide further details for this and the various projection types in App. B.3.

Center of mass shift The CoM-shift transform allows us to apply the aforementioned $\mathrm{SE}(3) \times S_n$ equivariant coupling in the ambient space rather than zero-COM subspace. In particular, before transforming our observed vector $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$ with $\mathcal{F}$, we lift it onto $\mathcal{X}$. We achieve this by swapping the center of mass between $\tilde{\boldsymbol{x}}$ and $\boldsymbol{a}$. For now, assume $\mathcal{A} = \mathcal{X}$, i.e. $k = 1$, with App. B.4 providing details for $k > 1$. Letting $\tilde{\mathcal{A}} \subseteq \mathcal{A}$ be the subspace where all augmented variables that differ by a translation occupy the same point, and $\tilde{\boldsymbol{a}} \in \tilde{\mathcal{A}}$ be defined analogously to $\tilde{\boldsymbol{x}}$, we apply the map $\mathrm{ShiftCoM} : \tilde{\mathcal{X}} \times \mathcal{A} \mapsto \mathcal{X} \times \tilde{\mathcal{A}}$ which acts on both of its arguments by subtracting from each of them the latter's CoM, that is,

$$\mathrm{ShiftCoM}(\tilde{\boldsymbol{x}}, \boldsymbol{a}) \triangleq (\tilde{\boldsymbol{x}} - \bar{\boldsymbol{a}},\, \boldsymbol{a} - \bar{\boldsymbol{a}}) \quad \text{with} \quad \bar{\boldsymbol{a}} \triangleq \tfrac{1}{n} \textstyle\sum_{i=1}^n [\boldsymbol{a}]^i. \tag{5}$$

This operation is invertible, with inverse $\mathrm{ShiftCoM}(\tilde{\boldsymbol{a}}, \boldsymbol{x})$, and has unit Jacobian determinant.

Algorithm 1: Flow block f

Inputs: Zero-CoM observation $\tilde{\boldsymbol{x}}$, augmented variable $\boldsymbol{a}$, Coupling transforms $\mathcal{F}_1, \mathcal{F}_2$

$(\boldsymbol{x}, \tilde{\boldsymbol{a}}) \leftarrow \text{ShiftCoM}(\tilde{\boldsymbol{x}}, \boldsymbol{a})$
$(\boldsymbol{x}, \tilde{\boldsymbol{a}}) \leftarrow \mathcal{F}_M^{(1)} \circ \cdots \circ \mathcal{F}_1^{(1)}(\boldsymbol{x}, \tilde{\boldsymbol{a}})$
$(\boldsymbol{a}, \tilde{\boldsymbol{x}}) \leftarrow \text{ShiftCoM}(\tilde{\boldsymbol{a}}, \boldsymbol{x})$
$(\boldsymbol{a}, \tilde{\boldsymbol{x}}) \leftarrow \mathcal{F}_M^{(2)} \circ \cdots \circ \mathcal{F}_1^{(2)}(\boldsymbol{a}, \tilde{\boldsymbol{x}})$

Output: $\tilde{\boldsymbol{x}}, \boldsymbol{a}$

Algorithm 2: Joint density evaluation

Inputs: $(\boldsymbol{x}, \boldsymbol{a}) \sim p$, base density q_0, $(f^{(l)})_{l=1}^L$
$(\tilde{\boldsymbol{x}}^{(0)}, \boldsymbol{a}^{(0)}) \leftarrow (\boldsymbol{x} - \bar{\boldsymbol{x}}, \boldsymbol{a} - \bar{\boldsymbol{x}})$
$\text{logdet} \leftarrow 0$
for $l = 1, \ldots, L$ **do**
  $(\tilde{\boldsymbol{x}}^{(l)}, \boldsymbol{a}^{(l)}) \leftarrow f^{(l)}(\tilde{\boldsymbol{x}}^{(l-1)}, \boldsymbol{a}^{(l-1)})$
  $\text{logdet} \leftarrow \text{logdet} + \left| \frac{\partial f^{(l)}(\tilde{\boldsymbol{x}}^{(l-1)}, \boldsymbol{a}^{(l-1)})}{\partial (\tilde{\boldsymbol{x}}^{(l-1)}, \boldsymbol{a}^{(l-1)})} \right|$

Output: $q_0(\tilde{\boldsymbol{x}}^{(L)}, \boldsymbol{a}^{(L)}) + \text{logdet}$

Putting the building blocks together Our flow transform is built as a sequence of L blocks. Each block, described in Alg. 1, consists of two equivariant coupling layers, see Fig. 1. Our observations $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$ are lifted onto $\mathcal{X}$ with ShiftCoM, they are transformed with M core transformations $\left(\mathcal{F}_i^{(1)}\right)_{i=1}^M$, and ShiftCoM is applied one more time to map the observations back to the zero-CoM hyperplane. After this, our augmented variables $\boldsymbol{a}$ are transformed with $\left(\mathcal{F}_i^{(2)}\right)_{i=1}^M$.

Joint density evaluation $q(\boldsymbol{x}, \boldsymbol{a})$ is performed with Alg. 2. We first subtract the observations' CoM from both the observed and augmented variables. We then apply our L flow transform blocks before evaluating the transformed variables' density under our base distribution q_0, which is described next. The log determinant of the core flow transform, f, has a contribution from the projection, transform in the invariant space, and inverse projection (see App. B.3 for details).

3.2 $\mathrm{SE}(3) \times S_n$ Invariant base distribution

Again, we assume $\mathcal{A} = \mathcal{X}$, i.e. $k = 1$, with the generalization given in App. B.4. Our invariant choice of base distribution is $q_0(\boldsymbol{x}, \boldsymbol{a}) = \tilde{\mathcal{N}}(\boldsymbol{x}; 0, I)\, \mathcal{N}(\boldsymbol{a}; \boldsymbol{x}, \eta^2 I)$ where $\boldsymbol{x} \in \mathbb{R}^{3n}$ and $\boldsymbol{a} \in \mathbb{R}^{3n}$ refer to $\boldsymbol{x}$ and $\boldsymbol{a}$ flattened into vectors, η^2 is a hyperparameter and we denote Gaussian distributions on $\tilde{\mathcal{X}}$ as $\tilde{\mathcal{N}}$ [Satorras et al., 2021a, Yim et al., 2023] with density

$$\tilde{\mathcal{N}}(\boldsymbol{x}; 0, I) = (2\pi)^{-3(n-1)/2} \exp(-\tfrac{1}{2}\|\tilde{\boldsymbol{x}}\|_2^2). \tag{6}$$

We sample from it by first sampling from a standard Gaussian $\mathcal{N}(0, I)$ and then removing the CoM. On the other hand, the distribution for $\boldsymbol{a}$ is supported on $\mathcal{A}$ which includes non-zero CoM points. It is centered on $\boldsymbol{x}$, yielding joint invariance to translations. The isotropic nature of q_0 makes its density invariant to rotations, reflections, and permutations [Satorras et al., 2021a, Yim et al., 2023].

3.3 Training and likelihood evaluation

In this section, we discuss learning and density evaluation with augmented variables.

Invariant augmented target distribution We assume the density of our observations p is $\mathrm{SE}(3) \times S_n$ invariant. Our target for augmented variables is $\pi(\boldsymbol{a}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{a}; \boldsymbol{x}, \eta^2 I)$, where η^2 matches the variance of the base Gaussian density over $\boldsymbol{a}$. This satisfies joint invariance $p(g \cdot \boldsymbol{x})\pi(g \cdot \boldsymbol{a}|g \cdot \boldsymbol{x}) = p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})$ for any $g \in \mathrm{SE}(3) \times S_n$, as shown in App. B.5.

Learning from samples When data samples $\boldsymbol{x} \sim p$ are available, we train our flow parameters by maximizing the joint likelihood, which is a lower bound on the marginal log-likelihood over observations up to a fixed constant

$$\mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}), \boldsymbol{a} \sim \pi(\boldsymbol{a}|\boldsymbol{x})}[\log q(\boldsymbol{x}, \boldsymbol{a})] \leq \mathbb{E}_{p(x)}[\log q(\boldsymbol{x})] + C.$$

Learning from energy When samples are not available but we can query the unnormalized energy of a state $U(\boldsymbol{x})$, with $p(\boldsymbol{x}) \propto \exp(-U(\boldsymbol{x}))$, we can minimize the joint reverse KL divergence. By the chain rule of the KL divergence, this upper bounds the KL between marginals

$$D_{\mathrm{KL}}\left(q(\boldsymbol{x}, \boldsymbol{a}) \,||\, p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x}))\right) \geq D_{\mathrm{KL}}\left(q(\boldsymbol{x}) \,||\, p(\boldsymbol{x})\right).$$

However, the reverse KL encourages mode-seeking [Minka, 2005] which may result in the model failing to characterize the full set of meta-stable molecular states. Therefore we instead use *flow annealed importance sampling bootstrap* (FAB) [Midgley et al., 2023], which targets the mass

covering α-divergence with $\alpha = 2$. In particular, we minimize the α-divergence over the joint which leads to an upper bound on the divergence of the marginals

$$D_2\left(q(\boldsymbol{x}, \boldsymbol{a}) \,||\, p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})\right) \triangleq \int \frac{p(\boldsymbol{x})^2 \pi(\boldsymbol{a}|\boldsymbol{x})^2}{q(\boldsymbol{x}, \boldsymbol{a})} \,\mathrm{d}\boldsymbol{a}\,\mathrm{d}\boldsymbol{x} \geq \int \frac{p(\boldsymbol{x})^2}{q(\boldsymbol{x})} \,\mathrm{d}\boldsymbol{x} \triangleq D_2\left(q(\boldsymbol{x}) \,||\, p(\boldsymbol{x})\right).$$

To compute unbiased expectations with the augmented flow we rely on the estimator $\mathbb{E}_{p(\boldsymbol{x})}[f(\boldsymbol{x})] = \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{a})}[w(\boldsymbol{x}, \boldsymbol{a})f(\boldsymbol{x})]$ where $w(\boldsymbol{x}, \boldsymbol{a}) = p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})/q(\boldsymbol{x}, \boldsymbol{a})$. Minimizing the joint α-divergence with $\alpha = 2$ corresponds to minimizing the variance in the joint importance sampling weights $w(\boldsymbol{x}, \boldsymbol{a})$, which allows for the aforementioned expectation to be approximated accurately.

Evaluating densities To evaluate the marginal density of observations we use the importance weighed estimator $q(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{a}\sim\pi(\cdot|\boldsymbol{x})}\left[\frac{q(\boldsymbol{x},\boldsymbol{a})}{\pi(\boldsymbol{a}|\boldsymbol{x})}\right]$, noting that π is Gaussian and thus supported everywhere. The estimator variance vanishes when $q(\boldsymbol{a}|\boldsymbol{x}) = \pi(\boldsymbol{a}|\boldsymbol{x})$, as shown in App. B.9.

4 Experiments

4.1 Training with samples: DW4, LJ13 and QM9 positional

First, we consider 3 problems that involving only positional information, with no additional features such as atom type or connectivity. Thus, the target densities are fully permutation invariant. The first two of these (DW4 and LJ13) are toy problems from Köhler et al. [2020], where samples are obtained by running MCMC on the 4 particle double well energy function (DW4) and 13 particles Leonard Jones energy function (LJ13) respectively. The third problem, QM9 positional [Satorras et al., 2021a] selects the subset of molecules with 19 atoms from the commonly used QM9 dataset [Ramakrishnan et al., 2014], and discards their node features.

For our model, Equivariant Augmented Coupling Flow (E-ACF), we consider all projection types (VECTOR-PROJ, CARTESIAN-PROJ, SPHERICAL-PROJ) and compare them to: (1) An augmented flow that is not rotation equivariant but is translation and permutation equivariant, as in [Klein et al., 2023]. We refer to this as NON-E-ACF. This model uses the same structure as the E-ACF but replaces the EGNN with a transformer which acts directly on atom positions, without any projection. We train NON-E-ACF with data-augmentation whereby we apply a random rotation to each sample within each training batch. (2) The SE(3) equivariant continuous normalizing flow (E-CNF) from Satorras et al. [2021a]. For QM9-positional, we found that the results reported by Satorras et al. [2021a] were based on a non-fully converged model, due to the large computational cost of training CNFs. Therefore, for QM9-postional we train an additional SE(3) equivariant diffusion mode [Hoogeboom et al., 2022] and evaluate it as a continuous normalizing flow (E-CNF-DIFF). All our generative models use the $\mathrm{SE}(3)$ GNN architecture proposed by Satorras et al. [2021b]. App. C.2.1 provides a detailed description of the configuration and architecture used within these experiments.

On Tab. 1, we see that on DW4 and LJ13, E-ACF is competitive with E-CNF. On QM9-positional the E-ACF clearly outperforms E-CNF-DIFF, while the under-trained E-CNF from Satorras et al. [2021a] performs poorly. The NON-E-ACF performs much worse than the E-ACF, despite being trained for more epochs, demonstrating the utility of in-build invariance. Furthermore, Fig. 2 shows that the distribution of inter-atomic distances of samples from our flow matches training data well. All E-ACF variants perform well except CARTESIAN-PROJ on QM9-positional, due to the numerical instability causing runs to crash early. Importantly, sampling and density evaluation of the E-ACF on an A100 GPU takes roughly 0.01 seconds. For the CNF trained by score matching (E-CNF-DIFF), sampling and density evaluation takes on average 5.05 seconds, with a standard deviation of 0.61 seconds due to inconsistency in the ODE solver. Thus the E-ACF is roughly than 500 times faster for sampling and density evaluation than the CNF.

4.2 Training with samples: Alanine dipeptide

Next, we approximate the Boltzmann distribution of alanine dipeptide in an implicit solvent at temperature $T = 800\,\mathrm{K}$. We train the models via maximum likelihood on samples generated by a replica exchange MD simulation [Mori and Okamoto, 2010], which serve as a ground truth. Our generated dataset consists of 10^6 samples in the training and validation set as well as 10^7 samples in the test set. Besides our E-ACF using the different projection schemes that we introduced in Sec. 3.1, we train the non-SO(3) equivariant flow (NON-E-ACF) with data augmentation similarly

Table 1: Negative log likelihood results on test set for flows trained by maximum likelihood. E-CNF results are from Satorras et al. [2021a]. On QM9-positional the CARTESIAN-PROJ E-ACF's training crashed due to numerical instability - results are reported for before the crash. Best results are emphasized in **bold**. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty.

	DW4	LJ13	QM9 positional
E-CNF	**8.15** $\pm N/A$	$30.56 \pm N/A$	$-70.2 \pm N/A$
E-CNF-DIFF	$N/A \pm N/A$	$N/A \pm N/A$	-128.50 ± 0.07
NON-E-ACF	10.69 ± 0.04	35.23 ± 0.15	-52.54 ± 2.74
VECTOR-PROJ E-ACF	9.08 ± 0.09	**30.21** $\pm$ **0.21**	-132.18 ± 1.15
CARTESIAN-PROJ E-ACF	8.89 ± 0.12	30.86 ± 0.18	-116.22 ± 14.11
SPHERICAL-PROJ E-ACF	8.71 ± 0.10	30.70 ± 0.11	**-136.31** $\pm$ **1.40**

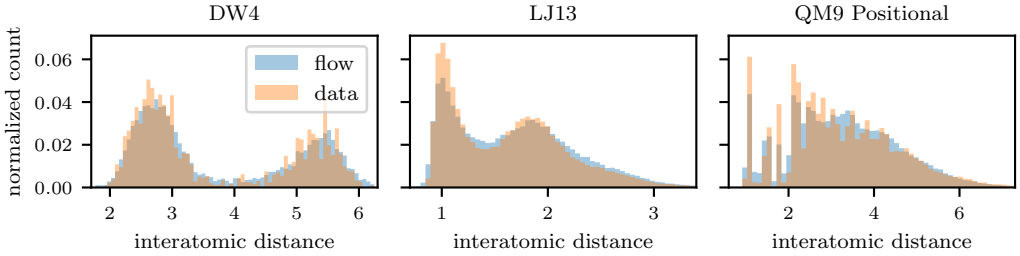


Figure 2: Inter-atomic distances for samples from the train-data and SPHERICAL-PROJ E-ACF.

to the previous experiments. Moreover, we train a flow on internal coordinates as in Midgley et al. [2023]. More details about the model architectures and training are given in App. C.2.2.

The results are shown in Fig. 3 and Tab. 2. All variants of our E-ACF clearly outperform the NON-E-ACF that is trained with data augmentation, as the Kullback Leibler divergence (KLD) of the Ramachandran plots is significantly lower and the negative log-likelihood (NLL) is higher. The flow trained on internal coordinates is only marginally better than our best model, i.e. the SPHERICAL-PROJ E-ACF. Note that it explicitly models the angles ϕ and ψ, while the E-ACF operates on the underlying Cartesian coordinates. To the best of our knowledge, our models are the first to learn the full Boltzmann distribution of a molecule purely on Cartesian coordinates while being competitive with a flow trained on internal coordinates.

4.3 Energy-based training: DW4 and LJ13

Lastly, we demonstrate that our proposed flow can be trained on the DW4 and LJ13 problems using only the target's unnormalized density with the FAB algorithm [Midgley et al., 2023]. The annealed importance sampling procedure within FAB requires sampling from the flow and evaluating its density multiple times. This is used within the training loop of FAB making it significantly more

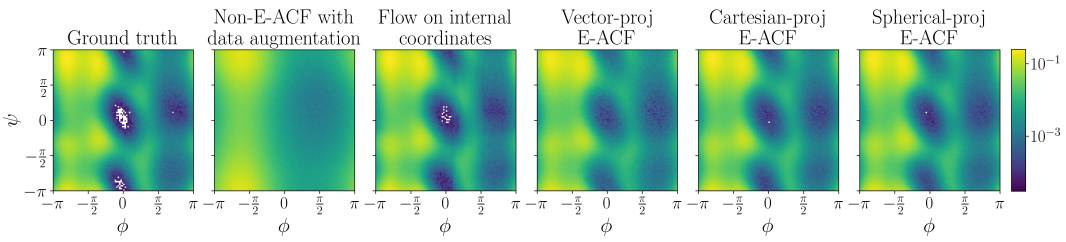


Figure 3: Ramachandran plots, i.e. marginal distribution of the dihedral angles ϕ and ψ (see App. C.2.2), obtained with MD (ground truth) and various normalizing flow models.

Table 2: KLD of the Ramachandran plots (see Fig. 3) produced by the models with respect to the ground truth and NLL on the test set. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty. Note that the flow trained on internal coordinates operates on a different manifold, and consequently the NLL value cannot be compared to the other models.

	KLD	NLL
Flow on internal coordinates	$(2.01 \pm 0.04) \cdot 10^{-3}$	(-179.57 ± 0.02)
NON-E-ACF	$(2.57 \pm 0.01) \cdot 10^{-1}$	-177.76 ± 0.78
VECTOR-PROJ E-ACF	$(1.46 \pm 0.22) \cdot 10^{-2}$	-188.50 ± 0.01
CARTESIAN-PROJ E-ACF	$(3.36 \pm 0.23) \cdot 10^{-3}$	$\mathbf{-188.60 \pm 0.00}$
SPHERICAL-PROJ E-ACF	$\mathbf{(2.77 \pm 0.30) \cdot 10^{-3}}$	$\mathbf{-188.61 \pm 0.01}$

expensive per parameter update than training by maximum likelihood. Given that sampling and density evaluation with CNFs is very expensive, training them with FAB is therefore intractable. Thus we only report results for our flow, as well as for the NON-E-ACF. We train the NON-E-ACF for more iterations than the E-ACF, such that the training times are similar, given that the NON-E-ACF is faster per iteration. App. C.3 provides a detailed description of the setup and computational resources used for these experiments.

Tab. 3 shows that the E-ACF trained with FAB successfully approximates the target Boltzmann distributions, with reasonably high joint effective sample size (ESS), and NLL comparable to the flows trained by maximum likelihood. Additionally, the ESS may be improved further by combining the trained flow with AIS, this is shown in App. C.3. In both problems the NON-E-ACF performs worse, both in terms of ESS and NLL. The flows trained by maximum likelihood have a much lower ESS (see App. C.2.1). This is expected, as unlike the $\alpha = 2$-divergence loss used in FAB, the maximum likelihood objective does not explicitly encourage minimizing importance weight variance. Furthermore, the flows trained by maximum likelihood use a relatively small, biased training set, which therefore limits their quality.

Table 3: Results for training by energy with FAB. Best results are emphasized in **bold**. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty.

	DW4		LJ13	
	Flow ESS (%)	NLL	Flow ESS (%)	NLL
NON-E-ACF	34.85 ± 0.98	7.48 ± 0.01	0.65 ± 0.20	37.43 ± 1.02
VECTOR-PROJ E-ACF	78.19 ± 0.33	7.15 ± 0.01	34.04 ± 1.47	30.66 ± 0.07
CARTESIAN-PROJ E-ACF	$\mathbf{80.07 \pm 0.72}$	7.16 ± 0.00	$\mathbf{38.93 \pm 2.35}$	30.47 ± 0.05
SPHERICAL-PROJ E-ACF	$\mathbf{79.50 \pm 0.94}$	$\mathbf{7.13 \pm 0.01}$	$\mathbf{39.37 \pm 0.89}$	$\mathbf{30.32 \pm 0.05}$

5 Discussion and Related Work

An alternative to our equivariant flow and equivariant CNFs are the equivariant residual flows proposed in Bose et al. [2022]. Alas, residual flows require fixed-point iteration for training and inversion. This is expensive and may interact poorly with methods like FAB [Midgley et al., 2023] which requires fast exact sampling and densities. Furthermore, Bose et al. [2022] found that the spectral normalization required for residual flows did not interact well with the equivariant CNN architecture in their experiments. Klein et al. [2023] proposed an augmented normalizing flow architecture to provide conditional proposal distributions for MD simulations and use a coupling scheme similar to ours. However, this model only achieves translation and permutation equivariance and the authors make their flow approximately rotation invariant through data augmentation. In our experiments, we found data augmentation to perform significantly worse than intrinsic invariance. In principle, Klein et al. [2023]'s model could be made fully invariant by substituting in our flow's projection-based coupling transform.

Our flow is compatible as a sub-module in flows modeling additional degrees of freedom. For example, our flow may be useful within a rigid body flow, proposed by Köhler et al. [2023b], which

operates on the relative position and orientation of a set of multiple rigid bodies. Specifically, to be made SE(3) equivariant, the center-of-masses of each body needs to be transformed using an SE(3) bijection, for which our flow may be used. Our implementation uses the E(3) equivariant EGNN proposed by Satorras et al. [2021a]. However, recently, there have been large efforts towards developing more expressive and efficient EGNNs architectures [Fuchs et al., 2020, Batatia et al., 2022, Liao and Smidt, 2023]. Incorporating these into our flow may improve performance and efficiency.

6 Conclusion

We have proposed an SE(3) equivariant augmented coupling flow that is achieves similar performance to CNFs when trained by maximum likelihood, while allowing for sampling and density evaluation more than 100 times faster. Furthermore, we showed that our flow can be trained as a Boltzmann generator using only the target's unnormalized density, on problems where internal coordinates are inadequate due to permutation symmetries, and doing so with a CNF is computationally intractable. It is possible to extend our model to learn the Boltzmann distribution of diverse molecules, by conditioning on their molecular graph, which we hope to explore in future work.

References

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.

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. *Nature communications*, 13(1):2453, 2022a.

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. *Nature communications*, 13, 2022b.

Avishek Joey Bose, Marcus Brubaker, and Ivan Kobyzev. Equivariant finite normalizing flows, 2022.

Andrew Campbell, Wenlong Chen, Vincent Stimper, Jose Miguel Hernandez-Lobato, and Yichuan Zhang. A gradient based strategy for Hamiltonian Monte Carlo hyperparameter optimization. In *Proceedings of the 38th International Conference on Machine Learning*, pages 1238–1248. PMLR, 2021.

Taco S. Cohen and Max Welling. Steerable cnns. *International Conference on Learning Representations*, 2016.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015. URL http://arxiv.org/abs/1410.8516.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7509–7520, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/7ac71d433f282034e088473244df8c02-Abstract.html.

Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks, 2020. URL https://arxiv.org/abs/2006.10503.

Mario Geiger and Tess E. Smidt. e3nn: Euclidean neural networks. *CoRR*, abs/2207.09453, 2022. doi: 10.48550/arXiv.2207.09453. URL https://doi.org/10.48550/arXiv.2207.09453.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 2022. URL https://proceedings.mlr.press/v162/hoogeboom22a.html.

Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. *arXiv preprint arXiv:2002.07101*, 2020.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi S. Jaakkola. Torsional diffusion for molecular conformer generation. *CoRR*, abs/2206.01729, 2022. doi: 10.48550/arXiv.2206.01729. URL https://doi.org/10.48550/arXiv.2206.01729.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Leon Klein, Andrew YK Foong, Tor Erlend Fjelde, Bruno Mlodozeniec, Marc Brockschmidt, Sebastian Nowozin, Frank Noé, and Ryota Tomioka. Timewarp: Transferable acceleration of molecular dynamics by learning time-coarsened dynamics. *arXiv preprint arXiv:2302.01170*, 2023.

Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: Exact likelihood generative learning for symmetric densities. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5361–5370. PMLR, 2020. URL http://proceedings.mlr.press/v119/kohler20a.html.

Jonas Köhler, Yaoyi Chen, Andreas Krämer, Cecilia Clementi, and Frank Noé. Flow-matching: Efficient coarse-graining of molecular dynamics without forces. *Journal of Chemical Theory and Computation*, 19(3):942–952, 2023a. doi: 10.1021/acs.jctc.3c00016.

Jonas Köhler, Michele Invernizzi, Pim de Haan, and Frank Noé. Rigid body flows for sampling molecular crystal structures. In *International Conference on Machine Learning*, 2023b.

Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=KwmPfARgOTD.

Yi Liu, Limei Wang, Meng Liu, Yuchao Lin, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d molecular graphs. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=givsRXsOt9r.

Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. *International Conference on Learning Representations*, 2023.

Tom Minka. Divergence measures and message passing. Technical report, Microsoft Research, 2005.

Yoshiharu Mori and Yuko Okamoto. Replica-exchange molecular dynamics simulations for various constant temperature algorithms. *Journal of the Physical Society of Japan*, 79(7):074001, 2010.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019. doi: 10.1126/science.aaw1147. URL https://www.science.org/doi/abs/10.1126/science.aaw1147.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Peter Petersen. *Linear Algebra*. Springer, 2012.

David Pollard. *A User's Guide to Measure Theoretic Probability*. Cambridge University Press, 2002.

Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

Danilo Jimenez Rezende, George Papamakarios, Sébastien Racanière, Michael S. Albergo, Gurtej Kanwar, Phiala E. Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 8083–8092. PMLR, 2020.

Victor Garcia Satorras, Emiel Hoogeboom, Fabian Fuchs, Ingmar Posner, and Max Welling. E(n) equivariant normalizing flows. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 4181–4192, 2021a. URL https://proceedings.neurips.cc/paper/2021/hash/21b5680d80f75a616096f2e791affac6-Abstract.html.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9323–9332. PMLR, 2021b. URL http://proceedings.mlr.press/v139/satorras21a.html.

Vincent Stimper, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Resampling Base Distributions of Normalizing Flows. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151, pages 4915–4936, 2022.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/41d80bfc327ef980528426fc810a6d7a-Abstract.html.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=JprM0p-q0Co.

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation. In *International Conference on Learning Representations*, 2022.

Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE(3) diffusion model with application to protein backbone generation, February 2023. URL http://arxiv.org/abs/2302.02277.

A Background

A.1 Tensors and representation

We assume a group $(G, \cdot)$, an algebraic structure that consists of a set G with a group operator $\cdot$ that follows the *closure*, *inverse* and *associativity* axioms.

A representation is an invertible linear transformation $\rho(g) : V \to V$, indexed by group elements $g \in G$, and defines how group elements g acts on elements of the vector space V. It satisfies the group structure $\rho(gh) = \rho(g)\rho(h)$.

A *tensor* T is a geometrical object that is acted on by group elements $g \in G$ in a particular way, characterised by its representation ρ: $g \cdot T = \rho(g)T$. Typical examples include scalar with $\rho_{\text{triv}}(g) = 1$ and vectors living in V with $\rho_{\text{Id}}(g) = g$. Higher order tensors which live in $V \times \cdots \times V$ can be constructed as $g \cdot T = (\rho_{\text{Id}}(g) \otimes \cdots \otimes \rho_{\text{Id}}(g))\,\text{vec}(T)$. A collection of tensors $(T_1, \ldots, T_n)$ with representations $(\rho_1, \ldots, \rho_n)$ can be stacked with fiber representation as the direct sum $\rho = \rho_1 \oplus \cdots \oplus \rho_n$.

A.2 Translational invariant densities

Proposition A.1 (Disintegration of measures for translation invariance [Pollard, 2002, Yim et al., 2023])**.** *Under mild assumptions, for every* $\text{SE}(3)$*-invariant measure* μ *on* $\mathbb{R}^{3\times n}$*, there exists* η *an* $\text{SO}(3)$*-invariant probability measure on* $\{\mathbb{R}^{3\times n} | \sum_{i=1}^n x_i = 0\}$ *and* $\bar{\mu}$ *proportional to the Lebesgue measure on* $\mathbb{R}^3$ *s.t.* $\mathrm{d}\mu([x_1, \ldots, x_n]) = \mathrm{d}\bar{\mu}(\frac{1}{n}\sum_{i=1}^n x_i) \times \mathrm{d}\eta([x_1 - \frac{1}{n}\sum_{i=1}^n x_i, \ldots, x_n - \frac{1}{n}\sum_{i=1}^n x_i])$.

The converse is also true, which motivates constructing an $\text{SO}(3)$-invariant probability measure on the subspace of $\mathbb{R}^{3\times n}$ with zero center of mass.

B Method

B.1 Transforming variables on a subspace

This section shows that a flow transform $\mathcal{F} : \mathcal{X} \to \mathcal{X}$ applied to observations $\tilde{\boldsymbol{x}}$ embedded in the zero-CoM subspace $\tilde{\mathcal{X}}$ does not have a well-defined Jacobian. We will show this by constructing such a transform, which leaves zero-CoM input unchanged, but for which the log-determinant is arbitrary.

Without loss of generality, we will assume in this subsection and the next that our observed and augmented variables are one dimensional, i.e., $\mathcal{X} = \mathcal{A} = \mathbb{R}^n$. Consider the orthonormal basis $V \in \mathbb{R}^{n\times n}$ constructed by concatenating orthonormal basis vectors $\boldsymbol{v}_i \in \mathbb{R}^n$ as $[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n]^T$. We choose $\boldsymbol{v}_1$ to be equal to $[1, 1, \ldots, 1]^T/\sqrt{n}$ such that inner products with $\boldsymbol{v}_1$ retrieve the CoM of a system of particles up to multiplication by a scalar: $\boldsymbol{v}_1^T\boldsymbol{x} = \sqrt{n}\bar{\boldsymbol{x}}$ and $\boldsymbol{v}_1^T\tilde{\boldsymbol{x}} = 0$. Let $\mathcal{F}(\boldsymbol{x}) = V^TSV\boldsymbol{x}$ be a mapping which rotates onto the basis defined by V, multiplies by the diagonal matrix $S = \text{diag}([s, 1, \ldots, 1])$ and then rotates back to the original space with V^T. This transformation only acts on the CoM. Applying this transformation to a zero-CoM variable $\tilde{\boldsymbol{x}}$ we have

$$\mathcal{F}(\tilde{\boldsymbol{x}}) = V^TSV\tilde{\boldsymbol{x}} = V^T[s\boldsymbol{v}_1^T\tilde{\boldsymbol{x}}, \boldsymbol{v}_2^T\tilde{\boldsymbol{x}}, \ldots, \boldsymbol{v}_n^T\tilde{\boldsymbol{x}}] = V^T[0, \boldsymbol{v}_2^T\tilde{\boldsymbol{x}}, \ldots, \boldsymbol{v}_n^T\tilde{\boldsymbol{x}}] = \tilde{\boldsymbol{x}},$$

leaving the zero CoM input unchanged. However, the log-determinant is

$$\log\left|\frac{\partial\mathcal{F}(\boldsymbol{x})}{\partial\boldsymbol{x}}\right| = \log\left|V^TSV\right| = \log|V^T| + \log|S| + \log|V| = \log s,$$

since for the unitary matrices $|V| = |V^T| = 1$. Because s is arbitrary, so is the density returned by any flow applying such a layer to zero-CoM variables. In the maximum likelihood setting, the aforementioned flow would lead to a degenerate solution of $s \to 0$ such that the log determinant would approach negative infinity and the log likelihood would approach infinity.

While our flow does use transforms which operate on non-CoM variables, we lift our CoM observations to the non-zero-CoM space before applying said transformations. We do this using the ShiftCoM operation, described in Sec. 3 and below.

B.2 Jacobian of ShiftCoM

We now show our $\mathrm{ShiftCoM} : \tilde{\mathcal{X}} \times \mathcal{A} \mapsto \mathcal{X} \times \tilde{\mathcal{A}}$ transformation to have unit Jacobian log determinant. We first, re-state the function definition

$$\mathrm{ShiftCoM}(\tilde{\boldsymbol{x}}, \boldsymbol{a}) \triangleq (\tilde{\boldsymbol{x}} - \bar{\boldsymbol{a}},\ \boldsymbol{a} - \bar{\boldsymbol{a}}) \quad \text{with} \quad \bar{\boldsymbol{a}} \triangleq \frac{1}{n}\sum_{i=1}^{n} [\boldsymbol{a}]^i.$$

We now re-write it using the orthonormal basis $V \in \mathbb{R}^{n\times n}$ defined in the previous subsection. For this, we use $I_2 \otimes V \in \mathbb{R}^{2n\times 2n}$ to refer to a 2×2 block diagonal matrix with V in both blocks, and we also stack the inputs and outputs of ShiftCoM, yielding $\mathrm{ShiftCoM}([\tilde{\boldsymbol{x}}, \boldsymbol{a}]) = (I_2 \otimes V)^T P (I_2 \otimes V)[\tilde{\boldsymbol{x}}, \boldsymbol{a}]$, where P is a permutation matrix that exchanges the first and $n+1$th elements of the vector it acts upon. It also flips this element's sign. To see this note

$$\begin{aligned}
\mathrm{ShiftCoM}([\tilde{\boldsymbol{x}}, \boldsymbol{a}]) &= (I_2 \otimes V)^T P (I_2 \otimes V)[\tilde{\boldsymbol{x}}, \boldsymbol{a}] \\
&= (I_2 \otimes V)^T P [0, \boldsymbol{v}_2^T\tilde{\boldsymbol{x}}, \dots, \boldsymbol{v}_n^T\tilde{\boldsymbol{x}}, \boldsymbol{v}_1^T\boldsymbol{a}, \dots, \boldsymbol{v}_n^T\boldsymbol{a}] \\
&= (I_2 \otimes V)^T [-\boldsymbol{v}_1^T\boldsymbol{a}, \boldsymbol{v}_2^T\tilde{\boldsymbol{x}}, \dots, \boldsymbol{v}_n^T\tilde{\boldsymbol{x}}, 0, \boldsymbol{v}_2^T\boldsymbol{a}, \dots, \boldsymbol{v}_n^T\boldsymbol{a}] \\
&= [\tilde{\boldsymbol{x}} - \bar{\boldsymbol{a}}, \boldsymbol{a} - \bar{\boldsymbol{a}}].
\end{aligned}$$

For the determinant we use that $|I_2| = 1$ and $|V| = 1$, and thus the determinant of their Kronecker product will be equal to one $|(I_2 \otimes V)| = 1$. Combining this with the fact that permutation matrices and matrices that flip signs have unit determinant, we arrive at

$$\begin{aligned}
\log \left| \frac{\partial \mathrm{ShiftCoM}([\tilde{\boldsymbol{x}}, \boldsymbol{a}])}{\partial [\tilde{\boldsymbol{x}}, \boldsymbol{a}]} \right| &= \log \left| (I_2 \otimes V)^T P (I_2 \otimes V) \right| \\
&= \log |(I_2 \otimes V)^T| + \log |P| + \log |I_2 \otimes V| = 1.
\end{aligned}$$

It is worth noting that ShiftCoM preserves the inherent dimensionality of its input; both its inputs and outputs consist of one zero-CoM vector and one non-zero-CoM vector. Because ShiftCoM does not apply any transformations to the CoM, it does not suffer from the issue of ill-definednes discussed in the previous subsection.

B.3 Projection variants and their respective transformations

In this section we provide a detailed description of the core transformation of our equivariant coupling flow layer, see Fig. 1. It consists of a projection into an invariant space via an equivariant reference, followed by a flow transformation in this space and a back-projection.

We denote the inputs with superscript ℓ and use $\ell+1$ for outputs to define our transformation block $\mathcal{F} : (\boldsymbol{x}^\ell; \boldsymbol{a}^\ell) \mapsto (\boldsymbol{x}^{\ell+1}, \boldsymbol{a}^{\ell+1})$. Without loss of generality, we assume that we are transforming $\boldsymbol{x}^\ell$. Then, $\mathcal{F}$ takes the following form.

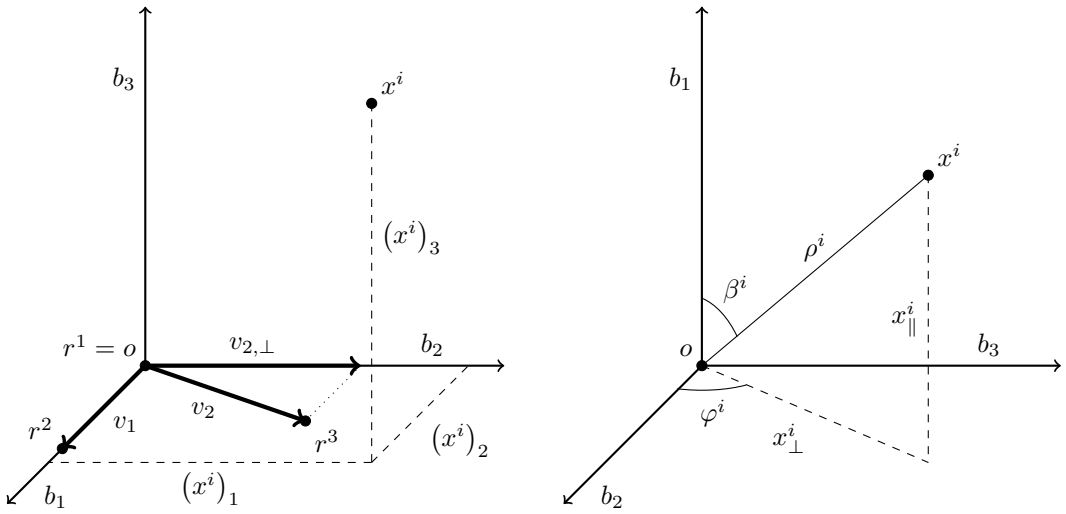
$$\begin{aligned}
\boldsymbol{x}^{\ell+1} &= \gamma_{\boldsymbol{r}}^{-1} \cdot \tau_\theta(\gamma_{\boldsymbol{r}} \cdot \boldsymbol{x}^\ell), \quad \text{with } (\boldsymbol{r}, \theta) = h(\boldsymbol{a}^\ell), \\
\boldsymbol{a}^{\ell+1} &= \boldsymbol{a}^\ell.
\end{aligned} \tag{7}$$

Here, h is a (graph) neural network that returns a set of u equivariant reference vectors $\boldsymbol{r} = \left[r^1, \dots, r^u\right]$, and invariant parameters θ. The equivariant vectors $\boldsymbol{r}$ parametrize $\gamma_{\boldsymbol{r}}$, a projection operator onto a rotation invariant feature space. This means that any rotations applied to the inputs will be cancelled by $\gamma_{\boldsymbol{r}}$, i.e. $\gamma_{g\cdot\boldsymbol{r}} = \gamma_{\boldsymbol{r}}^{-1} \cdot g^{-1}$ or equivalently $(\gamma_{g\cdot\boldsymbol{r}})^{-1} = g \cdot \gamma_{\boldsymbol{r}}^{-1}$ for all $g \in \mathrm{SO}(3)$.

In the next paragraphs, we discuss three projection operations and their respective flow transformations.

Cartesian projection Here, we predict a 3D Cartesian coordinate system as illustrated in Fig. 4a. The coordinates system has an origin $o = r^1$ and an orthonormal basis Q, both being equivariant. Q is constructed from two equivariant vectors $v_1 = r^2 - r^1$ and $v_2 = r^3 - r^1$ in the following way. To obtain the first basis vector, we simply normalize v_1, i.e. $b_1 = \frac{v_1}{\|v_1\|}$. We get the second basis vector via a Gram-Schmidt [Petersen, 2012] iteration, i.e. we compute the component of v_2 orthogonal to v_1, which is $v_{2,\perp} = v_2 - b_1^\top v_2$, and then normalize it to obtain $b_2 = \frac{v_{2,\perp}}{\|v_{2,\perp}\|}$. Note that this is ill-posed if v_1 and v_2 are collinear. To avoid this case, we introduce an auxiliary loss,

$$\text{aux-loss} = -\log\left(\epsilon + \arccos(\mathrm{abs}(v_1^T v_2 / \|v_1\| / \|v_2\|))\right), \tag{8}$$



(a) Cartesian projection.

(b) Spherical projection.

Figure 4: Illustration of the Cartesian and spherical projection.

which uses a logarithmic barrier function on the angle between the vectors to prevent them from being collinear. ϵ is a small constant that prevents the $\mathrm{aux\text{-}loss}$ from being equal to infinity. Finally, the third basis vector is $b_3 = b_1 \times b_2$, and we can assemble $Q = [b_1, b_2, b_3]$.

The projection into this coordinate system and the respective inverse operation are given by

$$\gamma_{\boldsymbol{r}}(\boldsymbol{x}) = Q^{-1}(\boldsymbol{x} - o)), \tag{9}$$

$$\gamma_{\boldsymbol{r}}^{-1}(\boldsymbol{x}_\mathrm{p}) = Q\boldsymbol{x}_\mathrm{p} + o, \tag{10}$$

where $\boldsymbol{x}_\mathrm{p}$ is the transformed projected variable.

In the projected space, we can apply any standard flow transform, as long as it acts on each particle individually. We choose τ_θ to an affine transformation as in the Real NVP architecture [Dinh et al., 2017].

Note that since b_3 is constructed via a cross product, it is not parity equivariant, which is why the entire transformation block is not parity equivariant either. However, since $\mathrm{sign}(b_3^\top r^2)$ is a pseudo-scalar, we can replace b_3 with $b_3' = \mathrm{sign}(b_3^\top r^2) b_3$, which is a vector, to retain parity equivariance.

The determinant of the Jacobian of $\mathcal{F}$ when using the Cartesian projection is just the determinant of the Jacobian of τ_θ, which we show below.

$$\begin{aligned}
\left|\frac{\partial \mathcal{F}(\boldsymbol{x};\boldsymbol{a})}{\partial(\boldsymbol{x},\boldsymbol{a})}\right| &= \left|\frac{\partial \mathcal{F}(\boldsymbol{x};\boldsymbol{a})|_{\boldsymbol{x}}}{\partial \boldsymbol{x}}\right| \cdot \left|\frac{\partial \mathcal{F}(\boldsymbol{x};\boldsymbol{a})|_{\boldsymbol{a}}}{\partial \boldsymbol{a}}\right| \\
&= \left|\left[\frac{\partial \gamma_{\boldsymbol{r}}^{-1}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right]_{\boldsymbol{z}=\tau_\theta(\gamma_{\boldsymbol{r}}(\boldsymbol{x}))} \left[\frac{\partial \tau_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}}\right]_{\boldsymbol{z}=\gamma_{\boldsymbol{r}}(\boldsymbol{x})} \frac{\partial \gamma_{\boldsymbol{r}}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right| \cdot 1 \\
&= \left|Q \left[\frac{\partial \tau_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}}\right]_{\boldsymbol{z}=\gamma_{\boldsymbol{r}}(\boldsymbol{x})} Q^{-1}\right| \\
&= |Q| \left|\left[\frac{\partial \tau_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}}\right]_{\boldsymbol{z}=\gamma_{\boldsymbol{r}}(\boldsymbol{x})}\right| |Q^{-1}| \\
&= \left|\left[\frac{\partial \tau_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}}\right]_{\boldsymbol{z}=\gamma_{\boldsymbol{r}}(\boldsymbol{x})}\right|.
\end{aligned}$$

Spherical projection For the spherical projection, which is illustrated in Fig. 4b, we compute an origin o and three Cartesian basis vectors b_1, b_2, and b_3 as in the Cartesian projection. Instead of

computing the Cartesian coordinates of each x^i, we compute the spherical coordinates (ρ, β, φ). Hence, for the i-th atom the projection operation and its inverse is given by

$$(\gamma_{\boldsymbol{r}}(\boldsymbol{x}))^i = \left(\|x^i\|, \arccos\left(\frac{x^i_{\|}}{\|x^i\|}\right), \operatorname{atan2}\left(b_3^\top x^i_\perp, b_2^\top x^i_\perp\right) \right)^\top, \tag{11}$$

$$\left(\gamma_{\boldsymbol{r}}^{-1}(\boldsymbol{x}_{\mathrm{p}})\right)^i = \rho^i \left(\cos(\beta^i) b_1 + \sin(\beta^i)\left(\cos(\varphi^i) b_2 + \sin(\varphi^i) b_3\right)\right), \tag{12}$$

Here, we used $x^i_{\|} = b_1^\top x^i$, $x^i_\perp = x^i - x^i_{\|}$, and $x^i_{\mathrm{p}} = (\rho^i, \beta^i, \varphi^i)$ for brevity.

The spherical coordinates are transformed with monotonic rational-quadratic splines [Durkan et al., 2019]. The radial coordinate ρ is transformed through a spline with the interval starting at 0 to ensure that the transformed coordinate is also non-negative to ensure invertibility. The two angles are transformed via circular splines [Rezende et al., 2020]. Similarly to the cartesian projection, we use the auxilliary loss (Eq. (8)) to prevent the vectors output from the GNN from being co-linear.

When using the spherical projection, we compute the log-determinant of the Jacobian of $\mathcal{F}$ by using the chain rule in the same way as we do for standard normalizing flows. The Jacobian determinant of the projection operation is given by the well-known term

$$\left| \frac{\partial \left(\gamma_{\boldsymbol{r}}(\boldsymbol{x})\right)^i}{\partial \boldsymbol{x}} \right| = \left(\rho^i\right)^2 \sin \beta^i. \tag{13}$$

For the inverse, we can just raise this term to the power of -1.

Vector projection This is a special case of the spherical projection, where we only transform the radial coordinate but leave the angles the same. Since in practice we only need to compute ρ, we call this case vector projection. Note that just predicting the origin o as a reference with the equivariant neural network is sufficient here. Unlike the cartesian and spherical projections, the vector projection does not rely on the cross product, and hence is parity equivariant by default.

B.4 Multiple augmented variables

Consider augmented variables consisting of multiple sets of observation-sized arrays, i.e. $\mathcal{A} = \mathcal{X}^k$ with $k > 1$. For the base distribution, and also the target, we deal with multiple augmented variables by letting them follow conditional normal distributions centered at $\boldsymbol{x}$. We thus have $\pi(\boldsymbol{a}|\boldsymbol{x}) = \prod_{i=1}^k \mathcal{N}(\boldsymbol{a}_i;\, \boldsymbol{x}, \sigma^2 I)$ and $q_0(\boldsymbol{x}, \boldsymbol{a}) = \tilde{\mathcal{N}}(\boldsymbol{x};\, 0, I) \prod_{i=1}^k \mathcal{N}(\boldsymbol{a}_i;\, \boldsymbol{x}, \sigma^2 I)$.

For our flow block, we group our variables into two sets $(\tilde{\boldsymbol{x}}, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_{(k-1)/2})$ and $(\boldsymbol{a}_{(k-1)/2+1}, \ldots, \boldsymbol{a}_k)$, and update one set conditional on the other, and vice-versa, while always ensuring the arrays being transformed have non-zero CoM.

Initially, $\boldsymbol{a}$ is non-zero-CoM with each augmented variable $\boldsymbol{a}_i$ having a CoM located at a different point in space. Before the first coupling transform $\mathcal{F}_M^{(1)}$, the $\mathrm{ShiftCoM}$ function applies the mapping

$$\mathrm{ShiftCoM}(\tilde{\boldsymbol{x}}, \boldsymbol{a}) \triangleq (\tilde{\boldsymbol{x}} - \bar{\boldsymbol{a}}_{1+\frac{k-1}{2}},\, \boldsymbol{a} - \bar{\boldsymbol{a}}_{1+\frac{k-1}{2}}) \quad \text{with} \quad \bar{\boldsymbol{a}}_{1+\frac{k-1}{2}} \triangleq \tfrac{1}{n} \textstyle\sum_{i=1}^n [\boldsymbol{a}_{1+\frac{k-1}{2}}]^i. \tag{14}$$

taking $\boldsymbol{a}_{(k-1)/2+1}$ to the zero-COM hyper-plane, while $\boldsymbol{x}$ and every other augmented variable remain non-zero-CoM. Before the second coupling transform $\mathcal{F}_M^{(2)}$, we apply the $\mathrm{ShiftCoM}$ maps in reverse, taking $\boldsymbol{x}$ back to the zero-COM hyper-plane.

The coupling transforms $\mathcal{F}_M^{(1)}$ and $\mathcal{F}_M^{(2)}$ work as previously, except now they transform $(k+1)/2$ sets of variables (holding the other set constant), where for each of the k variables, each of the n nodes get their own local reference frame and inner flow transform τ_θ.

B.5 Invariant base and target distributions

We now show the $\mathrm{SE}(3) \times S_n$ invariance of our extended base distribution choice, which trivial extends to our choice of augmented target distribution. Consider the choice $\pi(\boldsymbol{a}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{a};\, \boldsymbol{x}, \eta^2 I)$ for η^2 a scalar hyperparameter. With this choice, the constraint of interest is satisfied since the Isotropic Gaussian is rationally invariant and centered at $\boldsymbol{x}$. To see this, let $t \in \mathrm{SE}(3)$ be decomposed

into two components $t = (R, u)$ where $R \in \mathrm{SO}(3)$ is a 3×3 rotation matrix and $u \in \mathbb{R}^3$ represents a translation,

$$\begin{aligned}\mathcal{N}(R\boldsymbol{a} + u;\, R\boldsymbol{x} + u,\, \eta^2 I) &= \frac{1}{Z} \exp \frac{-1}{2}(R\boldsymbol{a} + u - R\boldsymbol{x} + u)^\top (\eta^{-2} I)(R\boldsymbol{a} + u - R\boldsymbol{x} + u) \quad (15)\\ &= \frac{1}{Z} \exp \frac{-1}{2}(\boldsymbol{a} - \boldsymbol{x})^\top R^\top (\eta^{-2} I) R(\boldsymbol{a} - \boldsymbol{x})\\ &= \frac{1}{Z} \exp \frac{-1}{2}(\boldsymbol{a} - \boldsymbol{x})^\top R^\top R(\eta^{-2} I)(\boldsymbol{a} - \boldsymbol{x})\\ &= \mathcal{N}(\boldsymbol{a};\, \boldsymbol{x},\, \eta^2 I) \quad (16)\end{aligned}$$

An analogous argument holds for permutations, since any $\sigma \in S_n$ can be represented as a unitary $n \times n$ matrix which is simultaneously applied to $\boldsymbol{a}$ and $\boldsymbol{x}$.

B.6 Maximum likelihood objective

We train our models from samples by maximising the joint log-density $\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}),\boldsymbol{a}\sim\pi(\boldsymbol{a}|\boldsymbol{x})}[\log q(\boldsymbol{x}, \boldsymbol{a})]$. This can be shown to be a lower bound on the expected marginal log-density over observations, which is the quantity targeted by models without augmented variables as

$$\mathrm{E}_{p(\boldsymbol{x})}[\log q(\boldsymbol{x})] = \mathrm{E}_{p(\boldsymbol{x})}\left[\log \mathrm{E}_{\pi(\boldsymbol{a}|\boldsymbol{x})}\left[\frac{q(\boldsymbol{x}, \boldsymbol{a})}{\pi(\boldsymbol{a}|\boldsymbol{x})}\right]\right] \geq \mathrm{E}_{p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})}\left[\log q(\boldsymbol{x}, \boldsymbol{a}) - \log \pi(\boldsymbol{a}|\boldsymbol{x})\right] \quad (17)$$

where we drop the latter term from our training objective since it is constant w.r.t. flow parameters. The bound becomes tight when $q(\boldsymbol{a}|\boldsymbol{x}) = \pi(\boldsymbol{a}|\boldsymbol{x})$.

B.7 Reverse KL objective

We now show the reverse KL objective involving the joint distribution of observations and augmented variables presented in the main text upper bounds the reverse KL over observations. Using the chain rule of the KL divergence we have

$$\begin{aligned}D_{\mathrm{KL}}\left(q(\boldsymbol{x}, \boldsymbol{a}) \,||\, p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})\right) &= D_{\mathrm{KL}}\left(q(\boldsymbol{x}) \,||\, p(\boldsymbol{x})\right) + \mathbb{E}_{q(\boldsymbol{x})} D_{\mathrm{KL}}\left(q(\boldsymbol{a}|\boldsymbol{x}) \,||\, \pi(\boldsymbol{a}|\boldsymbol{x})\right)\\ &\geq D_{\mathrm{KL}}\left(q(\boldsymbol{x}) \,||\, p(\boldsymbol{x})\right).\end{aligned}$$

The looseness in the bound is $\mathbb{E}_{q(\boldsymbol{x})} D_{\mathrm{KL}}\left(q(\boldsymbol{a}|\boldsymbol{x}) \,||\, \pi(\boldsymbol{a}|\boldsymbol{x})\right)$ which again vanishes when $q(\boldsymbol{a}|\boldsymbol{x}) = \pi(\boldsymbol{a}|\boldsymbol{x})$.

B.8 Alpha divergence objective

We now show that our joint objective represents an analogous upper bound for the α divergence when $\alpha = 2$, i.e., the objective targeted by FAB [Midgley et al., 2023]. Let $U(\boldsymbol{x})$ be the unnormalised energy function of interest evaluated at state $\boldsymbol{x}$, with $p(\boldsymbol{x}) \propto \exp(-U(\boldsymbol{x}))$. We have

$$\begin{aligned}D_2\left(q(\boldsymbol{x}) \,||\, p(\boldsymbol{x})\right) &= \int \frac{p(\boldsymbol{x})^2}{q(\boldsymbol{x})} \,\mathrm{d}\boldsymbol{x}\\ &\leq \int \frac{p(\boldsymbol{x})^2}{q(\boldsymbol{x})} \left(\int \frac{\pi(\boldsymbol{a}|\boldsymbol{x})^2}{q(\boldsymbol{a}|\boldsymbol{x})} \,\mathrm{d}\boldsymbol{a}\right) \mathrm{d}\boldsymbol{x}\\ &= \int \frac{p(\boldsymbol{x})^2 \pi(\boldsymbol{a}|\boldsymbol{x})^2}{q(\boldsymbol{x}) q(\boldsymbol{a}|\boldsymbol{x})} \,\mathrm{d}\boldsymbol{a}\,\mathrm{d}\boldsymbol{x}\\ &= D_2\left(q(\boldsymbol{x}, \boldsymbol{a}) \,||\, p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})\right)\end{aligned}$$

which also becomes tight when $q(\boldsymbol{a}|\boldsymbol{x}) = \pi(\boldsymbol{a}|\boldsymbol{x})$. The inequality is true because $\int \frac{\pi(\boldsymbol{a}|\boldsymbol{x})^2}{q(\boldsymbol{a}|\boldsymbol{x})} \,\mathrm{d}\boldsymbol{a} > 0$. To see this, take any densities π and q, and apply Jensen's inequality as

$$\int \frac{\pi(a)^2}{q(a)} \,\mathrm{d}a = \mathrm{E}_{\pi(a)} \frac{\pi(a)}{q(a)} = \mathrm{E}_{\pi(a)} \left(\frac{q(a)}{\pi(a)}\right)^{-1} \geq \left(\mathrm{E}_{\pi(a)} \frac{q(a)}{\pi(a)}\right)^{-1} = \int q(a) \mathrm{d}a = 1.$$

B.9 Importance weighted estimator

We define the following estimator $w := \frac{q(\boldsymbol{x},\boldsymbol{a})}{\pi(\boldsymbol{a}|\boldsymbol{x})}$ with $\boldsymbol{a} \sim \pi(\cdot|x)$. Assuming that whenever $\pi(\boldsymbol{a}|\boldsymbol{x}) = 0$ then $q(\boldsymbol{x}, \boldsymbol{a}) = 0$, we have that this estimator is trivially unbiased:

$$\begin{aligned}\mathbb{E}[w] &= \int \frac{q(\boldsymbol{x}, \boldsymbol{a})}{\pi(\boldsymbol{a}|\boldsymbol{x})} \pi(\boldsymbol{a}|\boldsymbol{x}) d\mu(\boldsymbol{a}) \\ &= \int q(\boldsymbol{x}, \boldsymbol{a}) d\mu(\boldsymbol{a}) \\ &= q(\boldsymbol{x})\end{aligned}$$

and that is variance is given by

$$\begin{aligned}\mathbb{V}[w] &= \mathbb{E}[(w - q(\boldsymbol{x}))^2] \\ &= \mathbb{E}[w^2] - \mathbb{E}[2wq(\boldsymbol{x})] + \mathbb{E}[q(\boldsymbol{x})^2] \\ &= \int \frac{q(\boldsymbol{x}, \boldsymbol{a})^2}{\pi(\boldsymbol{a}|\boldsymbol{x})^2} \pi(\boldsymbol{a}|\boldsymbol{x}) d\mu(\boldsymbol{a}) - 2q(\boldsymbol{x})\mathbb{E}[w] + q(\boldsymbol{x})^2 \\ &= \int \frac{q(\boldsymbol{x})q(\boldsymbol{a}|\boldsymbol{x})}{\pi(\boldsymbol{a}|\boldsymbol{x})} q(\boldsymbol{x}, \boldsymbol{a}) d\mu(\boldsymbol{a}) - q(\boldsymbol{x})^2\end{aligned}$$

thus if $q(\boldsymbol{a}|\boldsymbol{x}) = \pi(\boldsymbol{a}|\boldsymbol{x})$, $\mathbb{V}[w] = 0$.

The log marginal likelihood may therefore be estimated with

$$\log q(\boldsymbol{x}) \approx \log \frac{1}{M} \sum_{m=1}^{M} \frac{q(\boldsymbol{x}, \boldsymbol{a}_m)}{\pi(\boldsymbol{a}_m|\boldsymbol{x})} \quad \text{and} \quad \boldsymbol{a}_m \sim \pi(\cdot|\boldsymbol{x}), \tag{18}$$

which becomes accurate as M becomes large.

C Experiments: Details and Further Results

We provide the details of the hyper-parameters and compute used for each experiment. Additionally we provide the code *anonomized-for-review* for replicating our experiments. It also contains the configuration files with the exact hyperparameter for each model and training run.

C.1 Custom Optimizer

For all experiments we use a custom defined optimizer, which improves training stability by skipping updates with very large gradient norms, and clipping updates with a moderately large gradient norms. Our optimizer keeps track of a window the the last 100 gradient norms, and then for a given step: (1) If the gradient norm is more than 20 times greater than the median gradient norm within the window, then skip the gradient step without updating the parameters. (2) Otherwise, if the gradient norm is more than five times the median within the window then clip the gradient norm to five times the median. (3) Use the (possibly clipped) gradients to perform a parameter update using the adam optimizer [Kingma and Ba, 2014].

C.2 Training with samples

C.2.1 DW4, LJ13 and QM9 Positional

Experimental details Below we provide a description of the hyper-parameters used within each experiment. Additionally Tab. 4 provides the run-times for each experiment.

Datasets: For DW4 and LJ13 we use a training and test set of 1000 samples following Satorras et al. [2021a]. For QM9-positional we use a training, validation and test set of 13831, 2501 and 1,813 samples respectively also following Satorras et al. [2021a].

Flow experiment details All flow models using 12 blocks (see Alg. 1 for the definition of a block). For the noise scale of the augmented variables sampled from $\mathcal{N}(\boldsymbol{a}; \boldsymbol{x}, \eta^2 I)$, we use $\eta = 0.1$ for

both the base of the flow, and augmented target distribution. For the CARTESIAN-PROJ E-ACF and NON-E-ACF we use RealNVP [Dinh et al., 2017] for the flow transform, while for CARTESIAN-PROJ and VECTOR-PROJ we use a rational quadratic spline transform [Durkan et al., 2019]. For the spline we use 8 bins for all flows. We use 1 set of augmented variables for DW4 and LJ13, and 3 sets of augmented variables for QM9-positional. For all equivariant GNN's we implemented the GNN proposed by Satorras et al. [2021b] with 3 blocks. All MLP's have 2 layers, with a width of 32, 32 and 64 units for DW4, LJ13 and QM9-positional respectively. For the NON-E-ACF model we use a transformer with 3 layers of self-attention with 3, 4 and 8 heads for DW4, LJ13 and QM9-positional respectively. For DW4 and LJ13 we use the optimizer described in App. C.1 with a constant learning rate of 0.0002. For QM9-positional we use a cosine learning rate schedule, that is initialised to a value of 0.00006, peaks at 0.0002 after 10 epochs, and then decays back to a final learning rate of 0.00006. We use a batch size of 32 for all experiments, and 200, 256 and 256 epochs of the datasets for DW4, LJ13 and QM9-positional respectively. We train the NON-E-ACF for 4 times as many epochs for all problems, which results in roughly the same run time as the SPHERICAL-PROJ E-ACF. For the CARTESIAN-PROJ and SPHERICAL-PROJ E-ACF, we use a coefficient of 10. for the auxiliary loss (Eq. (8)), for weighting it relative to the maximum likelihood loss. Marginal log likelihood in Tab. 1 is calculated using Eq. (18) with 20 samples from the augmented variables $\boldsymbol{a}$ per test-set sample of $\boldsymbol{x}$.

Continuous normalizing flow / diffusion experiment details For the E-CNF we report the results from Satorras et al. [2021a]. Our E-CNF-DIFF implementation roughly follows Hoogeboom et al. [2022], but we use the noise schedule as Ho et al. [2020]. We use the same GNN architecture as for our flows, that is the E(n) GNN of [Satorras et al., 2021b], to predict the diffusion noise. The Network is composed of 3 message layers, each of which uses a 3 hidden layer MLP with 256 hidden units per layer. GNN expands its inputs to 256 scalars and 196 vectors per graph node and returns its predictions as a linear combination of the latter 196 vectors. This model has around 90M learnable parameters. We use the AdamW optimiser with a weight decay of 10^{-4} and run it for 5000 epochs. The initial learning rate is set to 10^{-5}. It is linearly increased to $5 \cdot 10^{-4}$ for the first 15% of training before being decreased to 10^{-6} with a cosine schedule. Following, Kingma et al. [2021], we sample noise timesteps during such that these are uniformly spaced within each batch, reducing gradient variance. We set the smallest accessible time value to 10^{-3}. For sampling and evaluating log-densities we use the `dopri5` ODE solver with a maximum number of evaluations of 1000.

Table 4: Run time for training by maximum likelihood on the DW4, LJ13 and QM9-positional datasets, in hours. All runs used an A100 GPU. Run times include evaluation that was performed intermittently throughout training. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty.

	DW4	LJ13	QM9 positional
E-CNF	$N/A \pm N/A$	$N/A \pm N/A$	$336 \pm N/A$
E-CNF-DIFF	$N/A \pm N/A$	$N/A \pm N/A$	3.80 ± 0.00
NON-E-ACF	0.23 ± 0.00	0.50 ± 0.01	8.91 ± 0.02
VECTOR-PROJ E-ACF	0.24 ± 0.00	0.48 ± 0.02	4.47 ± 0.02
CARTESIAN-PROJ E-ACF	0.20 ± 0.00	0.45 ± 0.02	3.53 ± 0.91
SPHERICAL-PROJ E-ACF	0.29 ± 0.00	0.83 ± 0.03	7.60 ± 0.15

Further results Tab. 5 provides the effective sample size of the flows trained by maximum likelihood, with respect to the joint target distribution $p(\boldsymbol{x})\pi(\boldsymbol{a}|\boldsymbol{x})$.

Table 5: Joint effective sample size evaluated using 10000 samples. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty.

	DW4	LJ13
NON-E-ACF	0.81 ± 0.50	0.05 ± 0.01
VECTOR-PROJ E-ACF	1.62 ± 0.40	0.10 ± 0.07
CARTESIAN-PROJ E-ACF	1.85 ± 0.43	0.06 ± 0.01
SPHERICAL-PROJ E-ACF	1.72 ± 0.40	0.14 ± 0.05

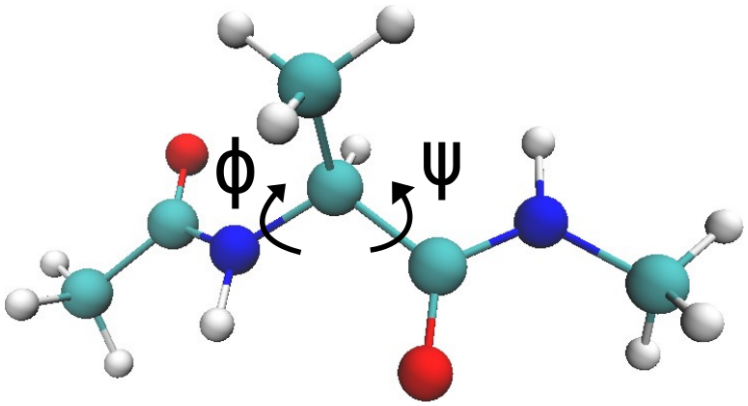


Figure 5: Visualization of the 22-atom molecule alanine dipeptide with its two dihedral angles ϕ and ψ.

C.2.2 Alanine dipeptide

The molecule alanine dipeptide and its dihedral angles ϕ and ψ, which are used in the Ramachandran plots in Fig. 3, are depicted in Fig. 5. We model it in an implicit solvent at a temperature of $T = 800\,\mathrm{K}$. The data used for maximum likelihood training and testing was generated with a replica exchange MD simulation [Mori and Okamoto, 2010] with 21 systems having temperatures from $300\,\mathrm{K}$ until $1300\,\mathrm{K}$ with $50\,\mathrm{K}$ temperature difference between each of them.

Each ANF has 16 layers with 4 core transformations per layer. The flow trained on internal coordinates has the same architecture as the models in Midgley et al. [2023], i.e. it is a coupling neural spline flow with 12 layers. The models were trained for 50 epochs.

Since the version of the VECTOR-PROJ flow is parity equivariant, the model learns to generate both chiral forms of alanine dipeptide [Midgley et al., 2023]. For evaluating the model, we filtered the samples from the model to only include those that correspond to the L-form, which is the one almost exclusively found in nature.

To plot and compute the KLD of the Ramachandran plots, we drew 10^7 samples from each model.

C.3 Energy-based training

For both the DW4 and LJ13 experiments we use an identical flow architecture and optimizer setup to the flows trained by maximum likelihood with samples App. C.2.1. For both experiments we use a batch size of 512. We train for 7502 and 6006 epochs for DW4 and LJ13 for the E-ACF. For the NON-E-ACF we train for 20002 and 40010 epochs for DW4 and LJ13 respectively, which corresponds to roughly the same training time as the SPHERICAL-PROJ E-ACF because the NON-E-ACF is faster per iteration. Evaluation of effective sample size is performed with 10,000 samples. **Hyper-parameters for FAB**: We set the minimum replay buffer size to 64 batches and maximum size to 128 batches. For the importance weight adjustment for re-weighting the buffer samples to account for the latest flow parameters, we clip to a maximum of 10. For AIS we use HMC with 5 leapfrog steps, and 2 and 8 intermediate distributions for DW4 and LJ13 respectively. We tune the step size of HMC to target an acceptance probability of 65%. We perform 8 buffer sampling and parameter update steps per AIS forward pass.

Table 6: Run time for energy based training with FAB on the DW4 and LJ13 problems. All runs used an A100 GPU. Run times include evaluation that was performed intermittently throughout training. The results are averaged over 3 seeded runs, with the standard error reported as uncertainty.

	DW4	LJ13
NON-E-ACF	4.9 ± 0.0	28.8 ± 0.1
VECTOR-PROJ E-ACF	3.6 ± 0.1	16.6 ± 0.0
CARTESIAN-PROJ E-ACF	3.2 ± 0.0	20.9 ± 0.0
SPHERICAL-PROJ E-ACF	4.6 ± 0.0	28.4 ± 0.1