malising flows are a family of distributions defined by a base distribution (usually a standard Gaussian), and a *diffeomorphism* f, *i.e.* a differentiable bijection with a differentiable inverse. Specifically, we set $p_\theta(x(t+\tau)|x(t))$ as the density of the distribution defined by the following generative process:

$$\begin{aligned} z^p, z^v &\sim \mathcal{N}(0, I) \\ x^p(t+\tau), x^v(t+\tau) &:= f_\theta(z^p, z^v; x^p(t), x^v(t)). \end{aligned} \tag{3}$$

Here $z^p \in \mathbb{R}^{3N}$ and $z^v \in \mathbb{R}^{3N}$. For all settings of θ and $x(t)$, $f_\theta(\,\cdot\,; x(t))$ is a diffeomorphism that takes the latent variables $(z^p, z^v) \in \mathbb{R}^{6N}$ to $(x^p(t+\tau), x^v(t+\tau)) \in \mathbb{R}^{6N}$. The conditioning state $x(t)$ parameterises a family of diffeomorphisms, defining a *conditional* normalising flow (Winkler et al., 2019). Note that there are no invertibility constraints on the mapping from the conditioning state $x(t)$ to the output $x(t+\tau)$, only the map from z to $x(t+\tau)$ must be invertible. Using the change of variables formula, we can evaluate $p_\theta(x(t+\tau)|x(t))$ analytically as:

$$\mathcal{N}\left(f_\theta^{-1}(x(t+\tau); x(t)); 0, I\right) \left|\det \mathcal{J}_{f_\theta^{-1}(\cdot; x(t))}(x(t+\tau))\right|,$$

where $f_\theta^{-1}(\,\cdot\,; x(t)) : \mathbb{R}^{6N} \to \mathbb{R}^{6N}$ is the inverse of the diffeomorphism $f_\theta(\cdot; x(t))$, and $\mathcal{J}_{f_\theta^{-1}(\cdot; x(t))}(x(t+\tau))$ denotes the Jacobian of $f_\theta^{-1}(\,\cdot\,; x(t))$ evaluated at $x(t+\tau)$.

3.2. Dataset generation

We now describe the dataset used to train the flow. We generate MD trajectories by integrating Equation (2) using the *OpenMM* library (Eastman et al., 2017). We simulate small proteins (peptides) in implicit water, *i.e.*, without explicitly modelling the degrees of freedom of the water molecules. Specifically, we generate a dataset of trajectories $\mathcal{D} = \{\mathcal{T}_i\}_{i=1}^P$, where P is the number of peptides. For each peptide i, we generate an MD trajectory that is temporally sampled with a spacing of τ, so that $\mathcal{T}_i = (x(0), x(\tau), x(2\tau), \ldots)$. During training, we randomly sample pairs $x(t), x(t+\tau)$ from $\mathcal{D}$. Each pair represents a sample from the conditional distribution $\mu(x(t+\tau)|x(t))$. These samples are used as examples to train the parameters θ of the flow. Additional details are provided in Appendix D. Since the flow is trained on trajectory data from *multiple* peptides, we can deploy it at test time to generalise to *new* peptides not seen in the training data.

3.3. Augmented normalising flows

Typically in molecular simulations, we are primarily interested in the distribution of the *positions* x^p, rather than the velocities x^v. Thus, it is not necessary for $x^v(t), x^v(t+\tau)$ to represent the actual velocities of the atoms in Equation (3). We hence simplify the learning problem by treating x^v as *non-physical auxiliary variables* within the *augmented normalising flow* framework (Huang et al., 2020). For each datapoint $x(t) = x^p(t), x^v(t)$ in $\mathcal{D}$, instead of obtaining $x^v(t)$ by recording the velocities in the MD trajectory, we *discard* the MD velocity and independently draw $x^v(t) \sim \mathcal{N}(0, I)$. The auxiliary variables $x^v(t)$ now contain no information about the future state $x^p(t+\tau), x^v(t+\tau)$, since $x^v(t)$ and $x^v(t+\tau)$ are drawn independently. Hence we can simplify f_θ to depend only on $x^p(t)$, with $x^p(t+\tau), x^v(t+\tau) := f_\theta(z^p, z^v; x^p(t))$. This raises the question of why auxiliary variables are necessary: we could instead directly model $p_\theta(x^p(t+\tau)|x^p(t))$, without the need for x^v. We include auxiliary variables for two reasons: First, they increase the expressivity of the distribution for x^p without a prohibitive increase in computational cost (Huang et al., 2020; Chen et al., 2020). Second, constructing a conditional flow that respects *permutation equivariance* is simplified with auxiliary variables — we discuss this in more detail in Section 4.1.

3.4. Targeting the Boltzmann distribution with MCMC

Once the flow $p_\theta(x(t+\tau)|x(t))$ has been trained, we use it as a proposal distribution in an MCMC method to target the joint distribution of the positions x^p and the auxiliary variables x^v, which has density:

$$\mu_{\text{aug}}(x^p, x^v) \propto \exp\left(-\tfrac{U(x^p)}{k_B T}\right) \mathcal{N}(x^v; 0, I). \tag{4}$$

Let m index the states in the Markov chain. Starting from an initial state $X_0 = (X_0^p, X_0^v) \in \mathbb{R}^{6N}$ at $m = 0$, we iterate:

$$\tilde{X}_m \sim p_\theta(\,\cdot\,|X_m^p) \tag{5}$$

$$X_{m+1} := \begin{cases} \tilde{X}_m & \text{with probability } \alpha(X_m, \tilde{X}_m) \\ X_m & \text{with probability } 1 - \alpha(X_m, \tilde{X}_m) \end{cases} \tag{6}$$

where $\alpha(X_m, \tilde{X}_m)$ is the *Metropolis-Hastings (MH) acceptance ratio* (Metropolis et al., 1953) targeting Equation (4):

$$\alpha(X, \tilde{X}) = \min\left(1, \frac{\mu_{\text{aug}}(\tilde{X}) p_\theta(X \mid \tilde{X}^p)}{\mu_{\text{aug}}(X) p_\theta(\tilde{X} \mid X^p)}\right) \tag{7}$$

The flow used for p_θ must allow for efficient sampling *and* exact likelihood evaluation, which is crucial for fast implementation of Equations (5) and (7). Additionally, after each MH step, we resample the auxiliary variables X^v using a *Gibbs sampling* update:

$$(X_m^p, X_m^v) \leftarrow (X_m^p, \epsilon), \quad \epsilon \sim \mathcal{N}(0, I). \tag{8}$$

Iterating these updates yields a sample $X_m^p, X_m^v \sim \mu_{\text{aug}}$ as $m \to \infty$. To obtain a Boltzmann-distributed sample of the positions $X_m^p \sim \mu$, we simply discard the auxiliary variables X_m^v. In practice, sending $m \to \infty$ is infeasible. Instead, we fix a computational budget and simulate the chain until the budget is reached. Algorithm 1 shows pseudocode for the MCMC procedure, using a *batch sampling* procedure for the proposals which significantly speeds up sampling.