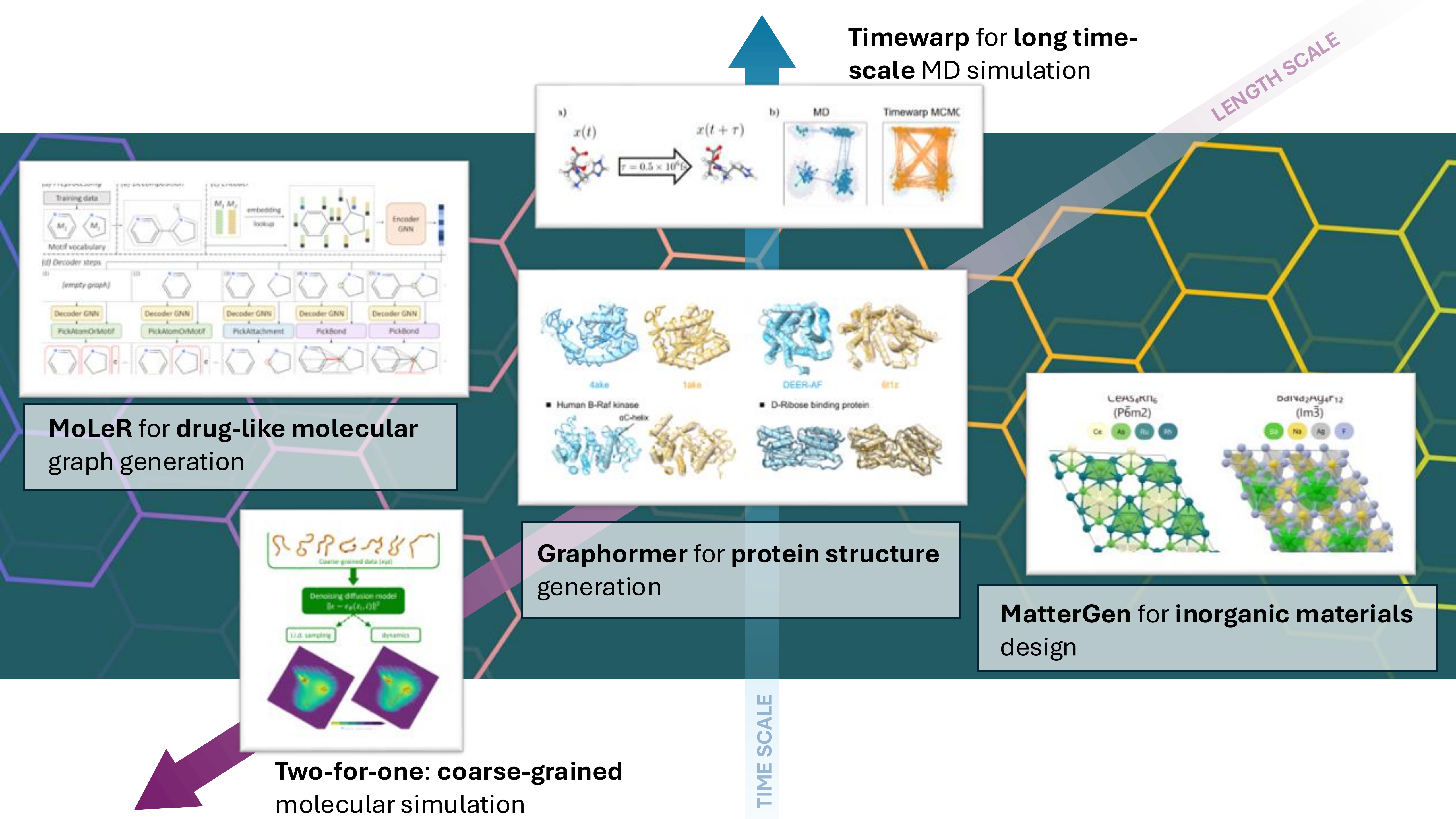
Diffusion Models

Chin-Wei Huang & Victor Garcia Satorras
Senior Researchers @ AI for Science – Microsoft Research Amsterdam

Microsoft Research AI for Science



Timewarp for long time-scale MD simulation
LENGTH SCALE
TIME SCALE
MoLeR for drug-like molecular graph generation
Graphormer for protein structure generation
MatterGen for inorganic materials design
Two-for-one: coarse-grained molecular simulation
MD
Timewarp MCMC
Human B-Raf kinase
D-Ribose binding protein
Denoising diffusion model

Outline

1. Introduction to Denoising Diffusion Models
2. Equivariant Diffusion Models for Molecule Generation in 3D
3. Workshop: Code & Practice

Slides borrowed from

Rianne van den Berg
Principle Research Manager
AI4Science, MSR
Amsterdam

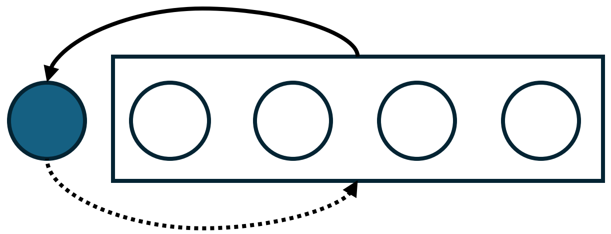
Daniel Zuegner
Senior Researcher
AI4Science, MSR
Berlin

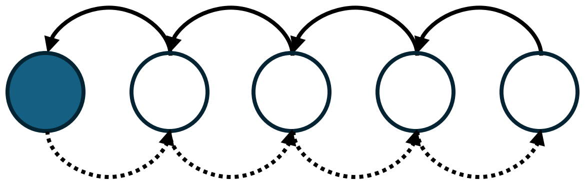
Sarah Lewis
Senior RSDE
AI4Science, MSR
Cambridge, UK

Outline

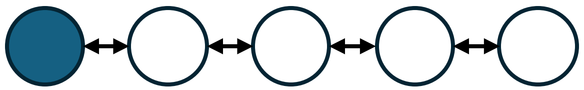
1. **Introduction to Denoising Diffusion Models**
2. Equivariant Diffusion Models for Molecule Generation in 3D
3. Workshop: Code & Practice

Very deep VAE / normalizing flows





Two networks p_θ and q_ϕ to optimize



Focus on data -> prior transformation

Major drawbacks of deep VAE / flow

- Space-time complexity grows with depth
- Arbitrary dynamics

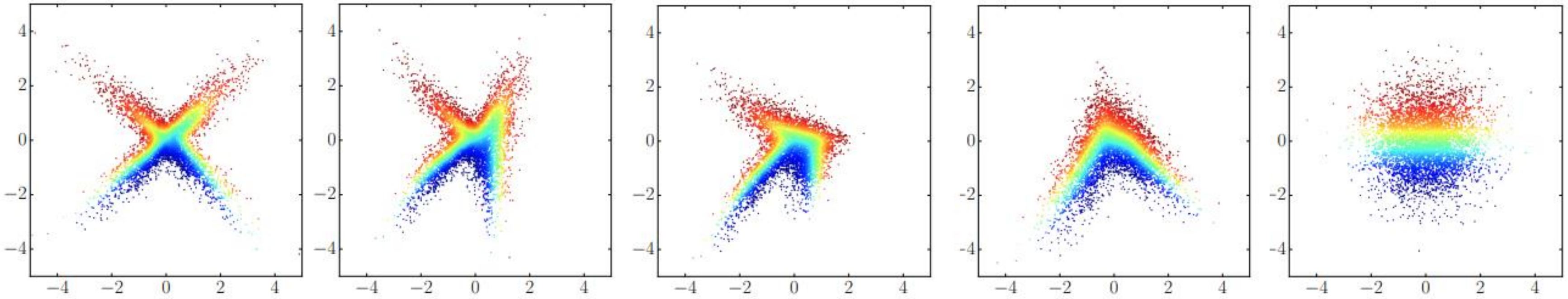
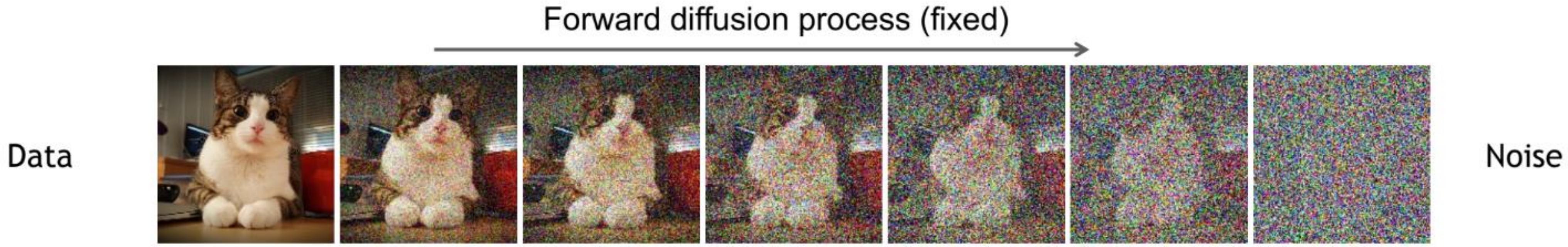


Image from Papamakarios et al JMLR 2021

What is a diffusion model?

Sohl-Dickstein et al., ICML 2015, Ho et al., NeurIPS 2020, Song et al., ICLR 2021

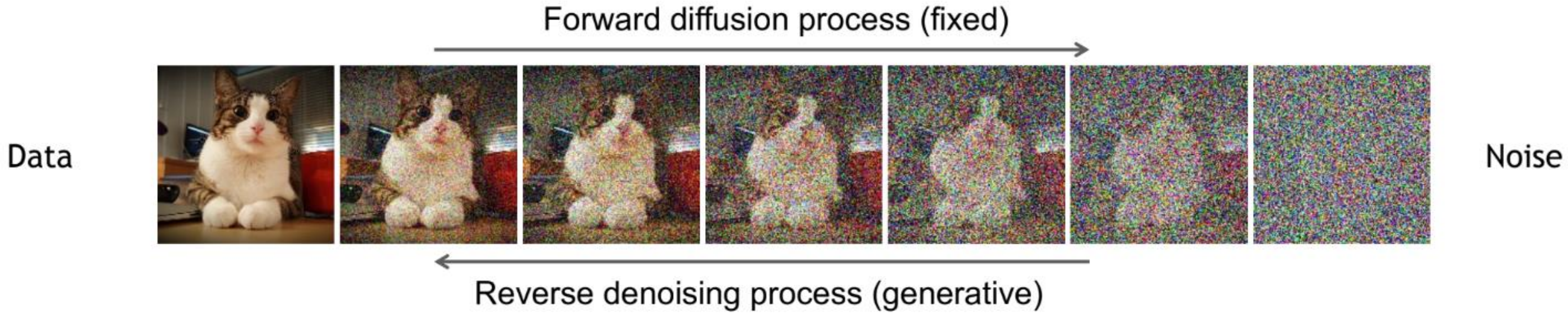
- Forward diffusion process gradually destroys information in the data.



What is a diffusion model?

Sohl-Dickstein et al., ICML 2015, Ho et al., NeurIPS 2020, Song et al., ICLR 2021

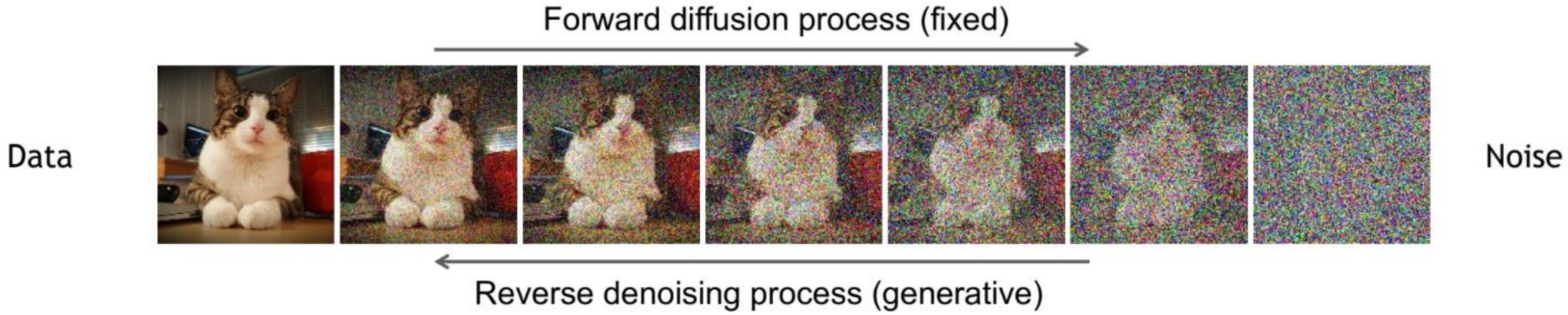
- Forward diffusion process gradually destroys information in the data.
- A generative model that learns to **revert** a **diffusion process**.



What is a diffusion model?

Sohl-Dickstein et al., ICML 2015, Ho et al., NeurIPS 2020, Song et al., ICLR 2021

- Forward diffusion process gradually destroys information in the data.
- A generative model that learns to **revert** a **diffusion process**.
- **Alternative perspectives** on diffusion models:
  - Deep generative models with latent variables & ELBO maximization
  - (Denoising) score matching



Recap: Latent Variable Generative Model

A latent (unobserved) random variable

$$p_\theta(x) = \int p_\theta(x, z)\,\mathrm{d}z$$

Goal: maximize $\mathbb{E}_{q(x)}[\log p_\theta(x)]$ & sample $x \sim p_\theta(x)$

Defining joint distribution as $p_\theta(x, z) = p_\theta(x|z)p(z)$ allows ancestral sampling of $(x, z) \sim p_\theta(x, z)$.

Evidence lower bound (ELBO)

$$\log p_\theta(x) = \log \int \dots \int p_\theta(x, z)\, dz$$

Evidence lower bound (ELBO)

$$
\begin{aligned}
\log p_\theta(x) &= \log \int \dots \int p_\theta(x, z)\, dz \\
&= \log \int \dots \int \frac{q(z|x)}{q(z|x)} p_\theta(x, z)\, dz \\
&= \log \mathbb{E}_{q(z_1,\dots,z_T|x)} \left[\frac{p_\theta(x, z)}{q(z|x)} \right] \\
&\geq \mathbb{E}_{q(z_1,\dots,z_T|x)} \log \left[\frac{p_\theta(x, z)}{q(z|x)} \right]
\end{aligned}
$$

Introducing (a lot of) latent variables

$$\begin{aligned}\log p_\theta(x) &= \log \int \dots \int p_\theta(x, z)\, dz \\ &= \log \int \dots \int \frac{q(z|x)}{q(z|x)} p_\theta(x, z)\, dz \\ &= \log \mathbb{E}_{q(z_1,\dots,z_T|x)} \left[\frac{p_\theta(x, z)}{q(z|x)}\right] \\ &\geq \mathbb{E}_{q(z_1,\dots,z_T|x)} \log \left[\frac{p_\theta(x, z)}{q(z|x)}\right]\end{aligned}$$

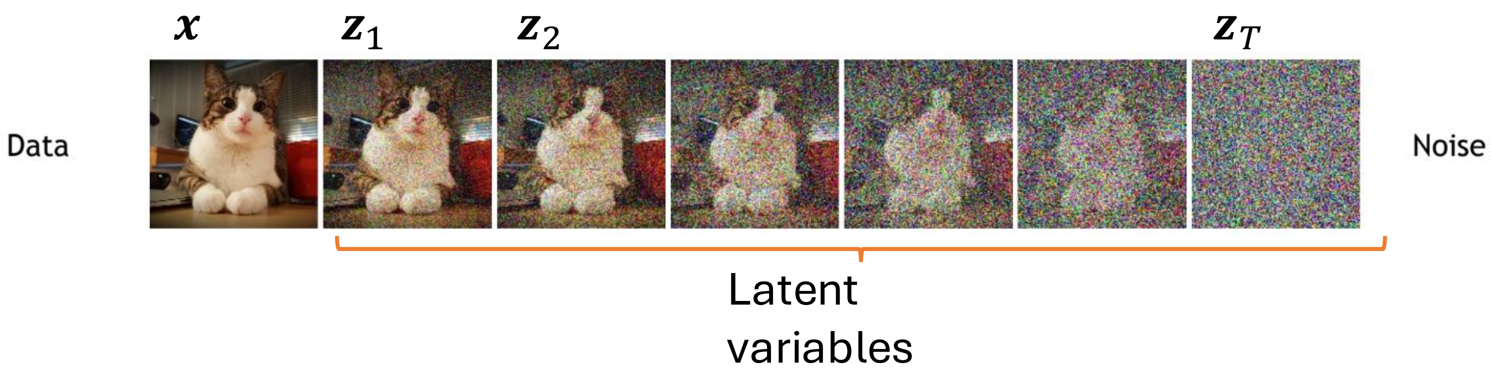
$$\begin{aligned}&= \log \int \dots \int p_\theta(x, z_1, \dots, z_T)\, dz_1 \dots dz_T \\ &= \log \int \dots \int \frac{q(z_1, \dots, z_T|x)}{q(z_1, \dots, z_T|x)} p_\theta(x, z_1, \dots, z_T)\, dz_1 \dots dz_T \\ &= \log \mathbb{E}_{q(z_1,\dots,z_T|x)} \left[\frac{p_\theta(x, z_1, \dots, z_T)}{q(z_1, \dots, z_T|x)}\right] \\ &\geq \mathbb{E}_{q(z_1,\dots,z_T|x)} \log \left[\frac{p_\theta(x, z_1, \dots, z_T)}{q(z_1, \dots, z_T|x)}\right]\end{aligned}$$

How to factorize $p_\theta(x, z_1, \dots, z_T)$ and $q(z_1, \dots, z_T|x)$?

How to factorize $q(z_1, \ldots, z_T | x)$ & $p_\theta(z_T, \ldots z_1, x)$?

Sohl-Dickstein et al., ICML 2015

(“Inference” of latent variables)



Forward: Define a *Markov* Chain to “infer” the latents given the data:

$$q(z_1, \ldots z_T | x) = q(z_1|x) q(z_2|z_1) \ldots q(z_T|z_{T-1})$$

Backward:

$$p_\theta(z_T, \ldots z_1, x) = p(z_T) p_\theta(z_{T-1}|z_T) \ldots p_\theta(z_1|z_2) p_\theta(x|z_1)$$

Organizing the negative ELBO

Sohl-Dickstein et al., ICML 2015, Ho et al., NeurIPS 2020, Song et al., ICLR 2021

Definitions of p and q:

$$p_\theta(z_T, \dots z_1, x) = p(z_T)p_\theta(z_{T-1}|z_T) \dots p_\theta(z_1|z_2)p_\theta(x|z_1)$$
$$q(z_1, \dots, z_T|x) = q(z_1|x)q(z_2|z_1) \dots q(z_T|z_{T-1})$$

Plugging it all in:

$$-\log p_\theta(x) \le -\mathbb{E}_{q(z_1,\dots,z_T|x)} \log\left[\frac{p_\theta(x, z_1, \dots, z_T)}{q(z_1, \dots, z_T|x)}\right]$$
$$= -\mathbb{E}_{q(z_1,\dots,z_T|x)} \log\left[\frac{p(z_T)p_\theta(z_{T-1}|z_T) \dots p(z_{t-1}|z_t) \dots p_\theta(z_1|z_2)p_\theta(x|z_1)}{q(z_T|x)q(z_{T-1}|z_T, x) \dots q(z_{t-1}|z_t, \dots, z_T, x) \dots q(z_1|z_2, \dots, z_T, x)}\right]$$
$$= -\mathbb{E}_{q(z_1,\dots,z_T|x)} \log\left[\frac{p(z_T)p_\theta(z_{T-1}|z_T) \dots p(z_{t-1}|z_t) \dots p_\theta(z_1|z_2)p_\theta(x|z_1)}{q(z_T|x)q(z_{T-1}|z_T, x) \dots q(z_{t-1}|z_t, x) \dots q(z_1|z_2, x)}\right]$$
$$= KL(q(z_T|x) \| p(z_T)) + \sum_{t=2}^{T} \mathbb{E}_{q(z_t|x)}[KL(q(z_{t-1}|z_t, x) \| p(z_{t-1}|z_t))] - \mathbb{E}_{q(z_1|x)}[\log p_\theta(x|z_1)]$$

Training diffusion models

$q(z_1, \dots z_T|x) = q(z_1|x)q(z_2|z_1) \dots q(z_T|z_{T-1})$

Data

Noise

$p_\theta(z_T, \dots z_1, x) = p(z_T)p_\theta(z_{T-1}|z_T) \dots p_\theta(z_1|z_2)p_\theta(x|z_1)$

$$L_{vb} = \mathbb{E}_{q(x)}\left[-\mathbb{E}_{q(z_1|x)} \log p_\theta\,(x|z_1)\right] + \sum_{t=2}^{T} \mathbb{E}_{q(z_t|x)}[KL[q(z_{t-1}|z_t, x)||p_\theta(z_{t-1}|z_t)] + KL[q(z_T|x)||p_\theta(z_T)]]$$

Practical requirements for q and p_θ to allow for efficient “simulation-free” training of p_θ:

1. Efficient sampling of z_t from $q(z_t|x)$ for arbitrary time t
2. Tractable expression for $q(z_{t-1}|z_t, x)$ (and the KL divergence).

For Gaussian or binomial $q(z_t|z_{t-1})$ (and $p_\theta(z_{t-1}|z_t)$):

Diffusion models with Gaussian distributions

Gaussian forward distributions: $q(z_t|z_{t-1}) = \mathcal{N}(z_t|\sqrt{\beta_t}z_{t-1}, (1-\beta_t)\mathrm{I})$ $(\beta_t < 1)$

1. Sampling arbitrary timesteps in one shot:

$$q(z_t|x) = \mathcal{N}\left(z_t|\sqrt{\bar{\alpha}_t}x, (1-\bar{\alpha}_t)\mathrm{I}\right)$$

$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

2. Tractable expression for posterior:

$$q(z_{t-1}|z_t, x) = \mathcal{N}\left(z_{t-1}|\tilde{\mu}_t(z_t, x), \tilde{\beta}_t\mathrm{I}\right)$$

$$\tilde{\mu}_t(z_t, x) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}z_t \qquad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

Additional guarantees

Gaussian KL has closed-form

$$-\log p_\theta(x) \leq -\mathbb{E}_{q(z_1|x)} \log p_\theta(x|z_1) + \sum_{t=2}^{T} \mathbb{E}_{q(z_t|x)}[KL[q(z_{t-1}|z_t, x)||p_\theta(z_{t-1}|z_t)]] + \cancel{KL[q(z_T|x)||p(z_T)]}$$

When $T \to \infty$:

1. For many choices of the noise schedule, we get a stationary distribution $\lim_{T' \to \infty} q(z_{T'}|x) = N(0, I)$.

   → pick $p(z_T) = N(0, I)$

2. As $T \to \infty$, the optimal reverse (generative) becomes Gaussian-like.

   → $p_\theta(z_{t-1}|z_t) = \mathcal{N}(z_{t-1}|\mu_\theta(z_t, t), \sigma_t I)$

Parameterizing $\mu_\theta(z_t, t)$

Ho et al., NeurIPS 2020

$$\mathbb{E}_{q(z_t|x)}[KL[q(z_{t-1}|z_t,x)||p_\theta(z_{t-1}|z_t)] = \mathbb{E}_{\boxed{q(z_t|x)}}\left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(z_t,x) - \mu_\theta(z_t,t)||^2\right] + C$$

Simplest option: $\mu_\theta(z_t,t) = \mathrm{nn}_\theta(z_t,t)$

Recall:

$$\boxed{z_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}\epsilon}$$

Alternative:

$$\tilde{\mu}_t(z_t,x) = \frac{1}{\sqrt{\alpha_t}}\left(z_t(x,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

Predict the noise:

$$\mu_\theta(z_t,t) = \frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boxed{\epsilon_\theta(z_t,t)}\right)$$

Loss, noise-scheduling, training and sampling

Ho et al., NeurIPS 2020

$$L_{vlb} = \mathbb{E}_{q(x)}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\mathbb{E}_{t\sim U(2,T)}\left[\underbrace{\boxed{\frac{\beta_t^2}{2\sigma_t^2\alpha_t\,(1-\bar{\alpha}_t)}}}_{\lambda_t}\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\boldsymbol{x}+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t\right)\right\|_2^2\right] + \text{reconstruct}$$

Ho et al. 2020 found that setting $\lambda_t = 1$ improves sample quality, i.e., the training loss is:

$$L_{simple} = \mathbb{E}_{q(x)}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\mathbb{E}_{t\sim U(2,T)}\left[\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\boldsymbol{x}+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t\right)\right\|_2^2\right] + \text{reconstruct}$$

Algorithm 1 Training

1: **repeat**
2: $\quad \boldsymbol{x} \sim q(\boldsymbol{x})$
3: $\quad t \sim \mathrm{Uniform}(\{1,\dots,T\})$
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
5: $\quad$ Take gradient descent step on
$$\nabla_\theta \left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\sqrt{\alpha_t}\,\boldsymbol{x}+\sqrt{1-\alpha_t}\,\boldsymbol{\epsilon},t)\right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
2: **for** $t = T,\dots,1$ **do**
3: $\quad \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ if $t > 1$, else $\epsilon = \mathbf{0}$
4: $\quad \mathbf{z}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\boldsymbol{\epsilon}_\theta(\mathbf{z}_t,t)\right) + \sigma_t\epsilon$
5: **end for**
6: **return** $\mathbf{z}_0$

Results from Ho et al. 2020

Sohl-Dickstein et al., ICML 2015, Ho et al., NeurIPS 2020

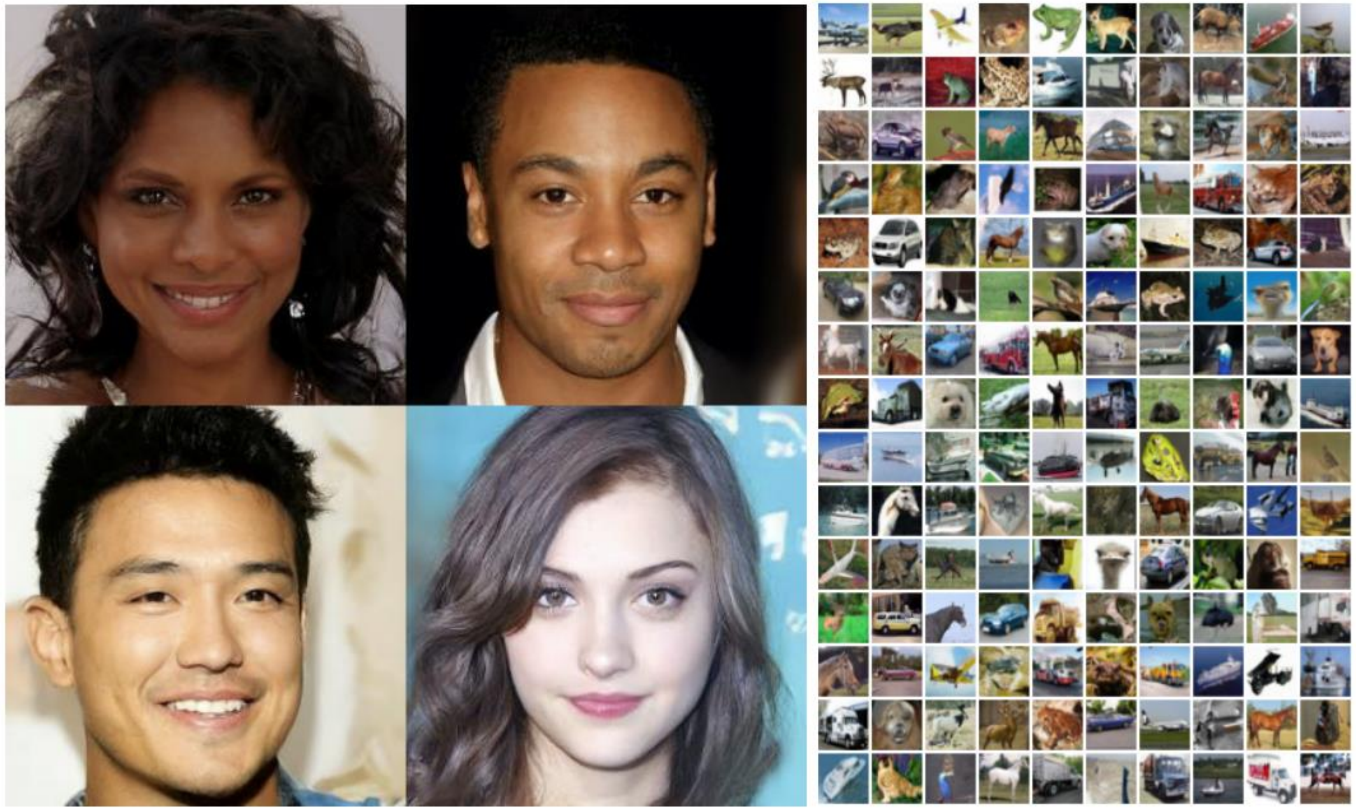


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	**10.06**	**2.67**	
Unconditional			
Diffusion (original) [53]			≤ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			**2.80**
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87±0.12	25.32	
SNGAN [39]	8.22±0.05	21.7	
SNGAN-DDLS [4]	9.09±0.10	15.42	
StyleGAN2 + ADA (v1) [29]	**9.74** ± 0.05	3.26	
Ours (L, fixed isotropic $\mathbf{\Sigma}$)	7.67±0.13	13.51	≤ 3.70 (3.69)
Ours (L_{simple})	9.46±0.11	**3.17**	≤ 3.75 (3.72)

Major drawbacks of deep VAE / flow

- Space-time complexity grows with depth

→ Simple choice of $q(z_{t-1}|z_t)$ which allows for simulation-free training since we can directly sample from $q(z_{t-1}|x)$

- Arbitrary dynamics

→ Fix $q(z_{t-1}|z_t)$ during training, and only train the decoder to revert the noise-injecting dynamics

Backup slides

Connection to denoising score matching

$$\nabla_\theta \mathbb{E}_{z_t}\left[\left\|s_\theta(z_t,t) - \nabla_{z_t}\log q(z_t)\right\|^2\right]$$

$$= \nabla_\theta \mathbb{E}_{z_t,x}\left[\left\|s_\theta(z_t,t) - \nabla_{z_t}\log q(z_t|x)\ \right\|^2\right]$$

$$\propto \nabla_\theta \mathbb{E}_{\epsilon,x}\left[\|\epsilon_\theta(z_t,t) - \epsilon\|^2\right]$$

set $s_\theta(\cdot,t) = -\frac{\epsilon_\theta(\cdot,t)}{\sqrt{1-\overline{\alpha}_t}}$

$$\boxed{z_t = \sqrt{\overline{\alpha}_t}x + \sqrt{1-\overline{\alpha}_t}\epsilon}$$

Algorithm 2 Sampling

1: $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
2: **for** $t = T,\dots,1$ **do**
3: $\quad\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ if $t>1$, else $\epsilon = \mathbf{0}$
4: $\quad\mathbf{z}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta(\mathbf{z}_t,t)\right) + \sigma_t\epsilon$
5: **end for**
6: **return** $\mathbf{z}_0$

$$z_t' \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left(z_t + \sqrt{\beta_t}\sqrt{1-\overline{\alpha}_t}\,s_\theta(z_t,t)\right) + \sigma_t\epsilon$$

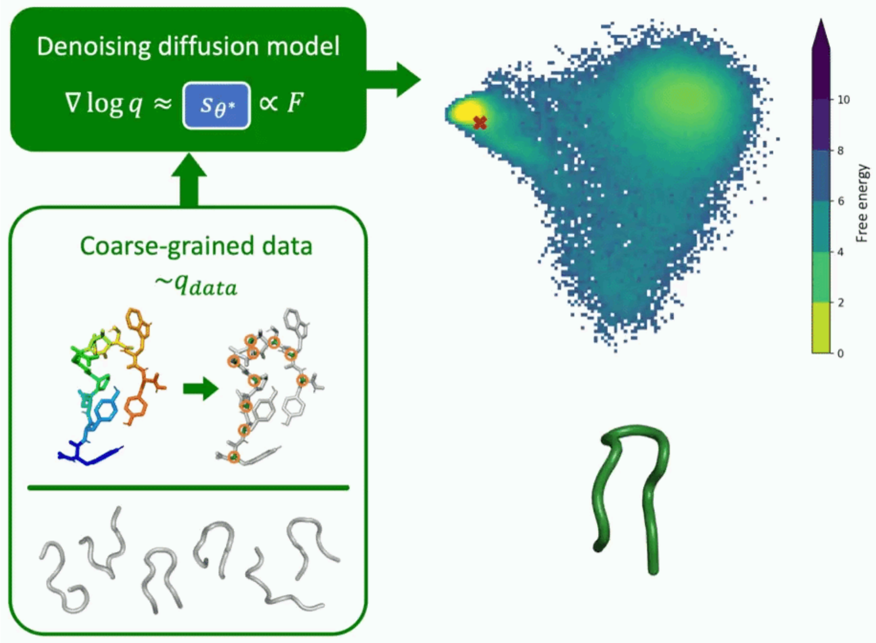
$$z_t' \leftarrow z + \gamma\nabla_z\log q(z_t) + \sqrt{2\gamma}\epsilon$$ (underdamped Langevin dynamics)

What can we use the score for

- Energy is composable -> classifier-based / -free guidance
- Accelerated sampling or corrector
- Extracted forces can be used for molecular dynamics

Two for One: Diffusion Models and Force Fields for Coarse-Grained Molecular Dynamics

Marloes Arts,[*,†,‡,@] Victor Garcia Satorras,[*,¶,@] Chin-Wei Huang,[¶] Daniel Zügner,[§] Marco Federici,[†,∥] Cecilia Clementi,[§,⊥] Frank Noé,[§] Robert Pinsler,[#] and Rianne van den Berg[¶]

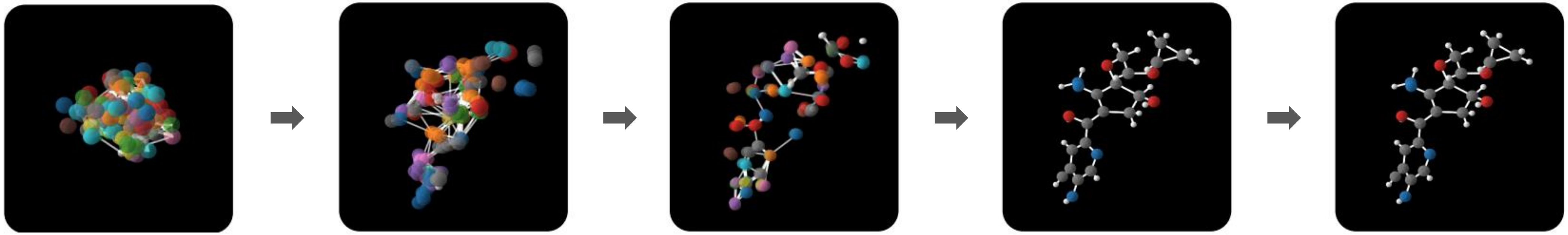


Outline

1. Introduction to Denoising Diffusion Models
2. **Equivariant Diffusion Models for Molecule Generation in 3D**
3. Workshop: Code & Practice

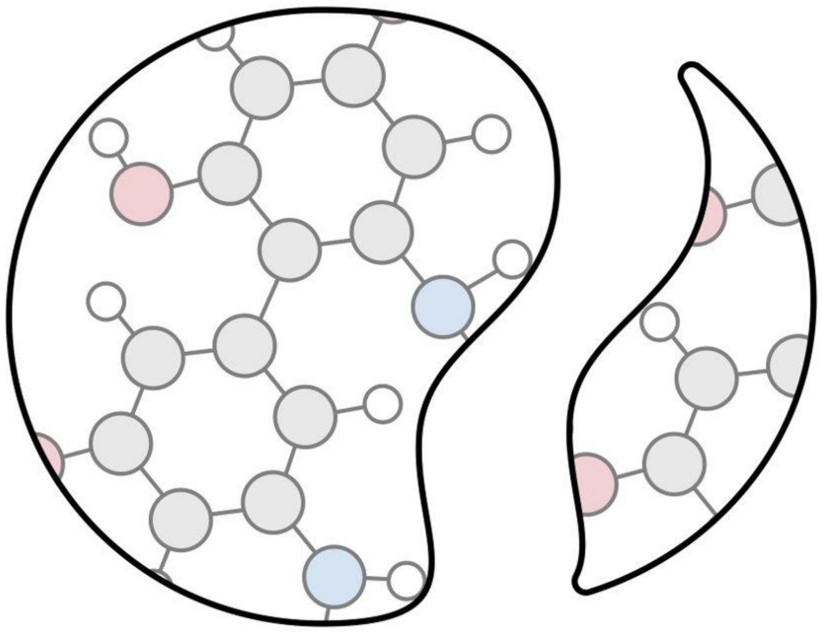
Equivariant Diffusion for Molecule Generation in 3D

Emiel Hoogeboom *[1] Victor Garcia Satorras *[1] Clément Vignac *[2] Max Welling [1]



Why Generate Molecules in 3D?

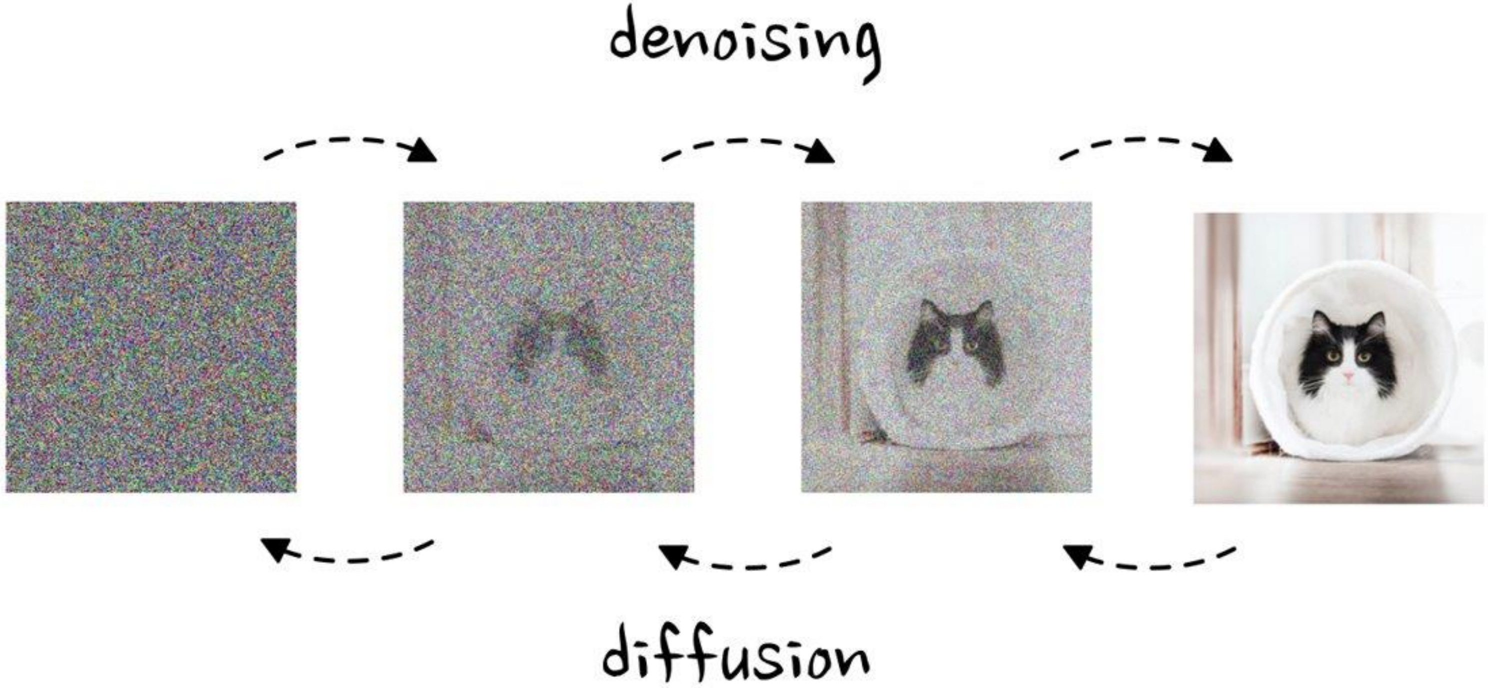
- Drug Discovery
- Catalyst Design
- Material Discovery
- Docking Problems



Background: Denoising Diffusion Process

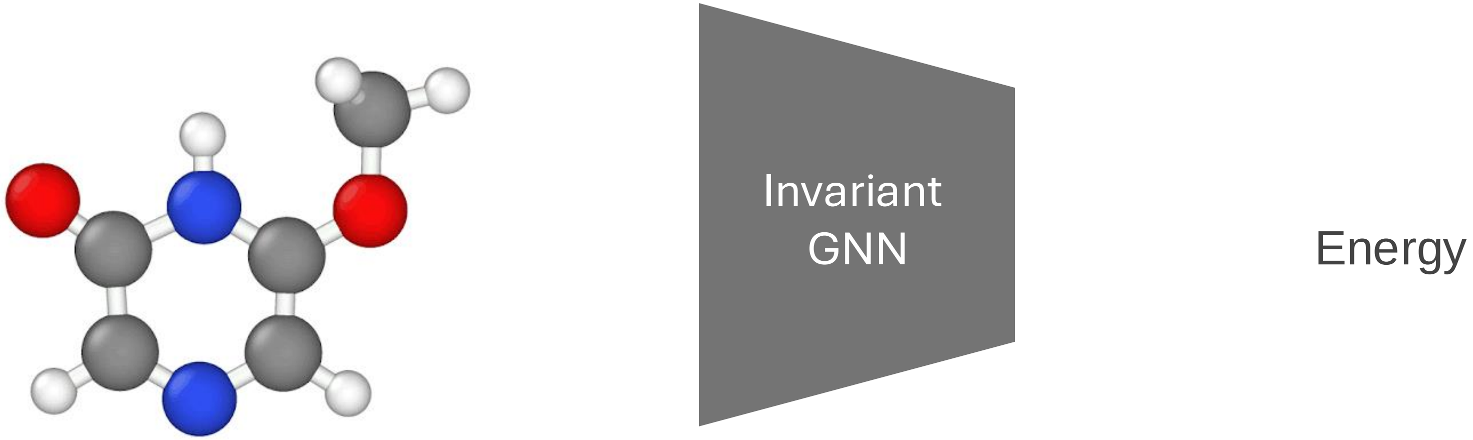
Diffuse: Destroy signal towards Normal Gaussian.

Denoise: Learn a denoising process to generate.



Background: Equivariance

Invariance / Equivariance



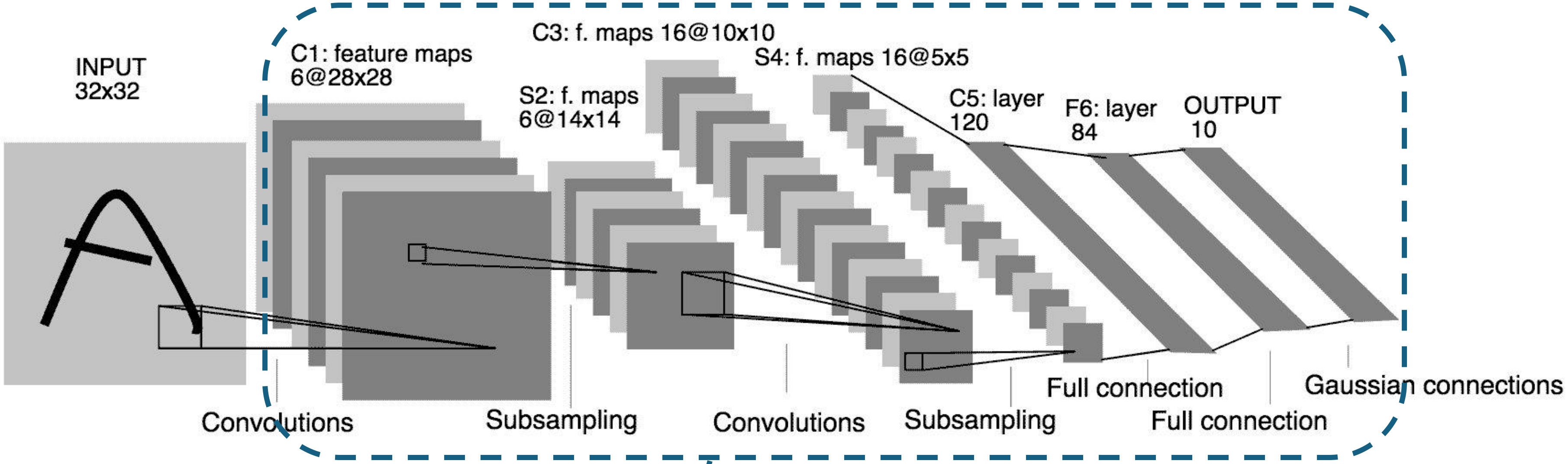
$$f(\boldsymbol{x}) = f(\mathbf{R}\boldsymbol{x})$$

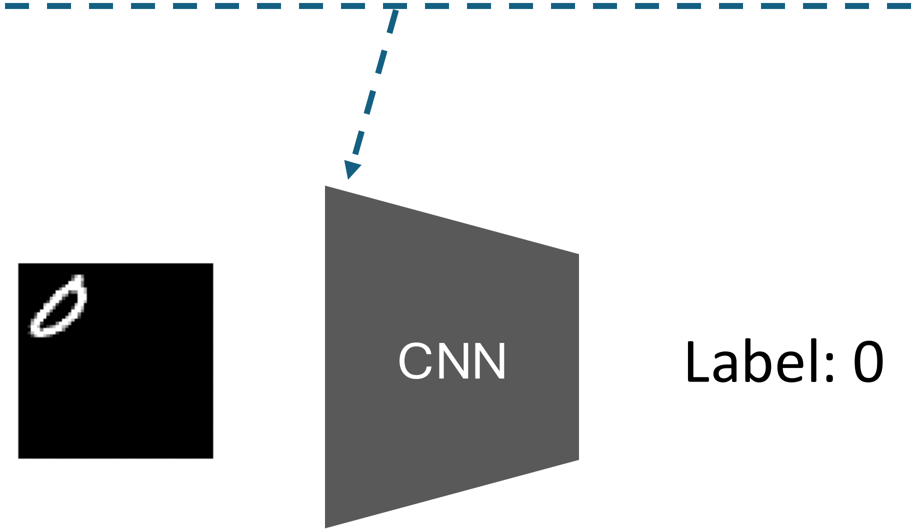
Background: Equivariance

Invariance / Equivariance

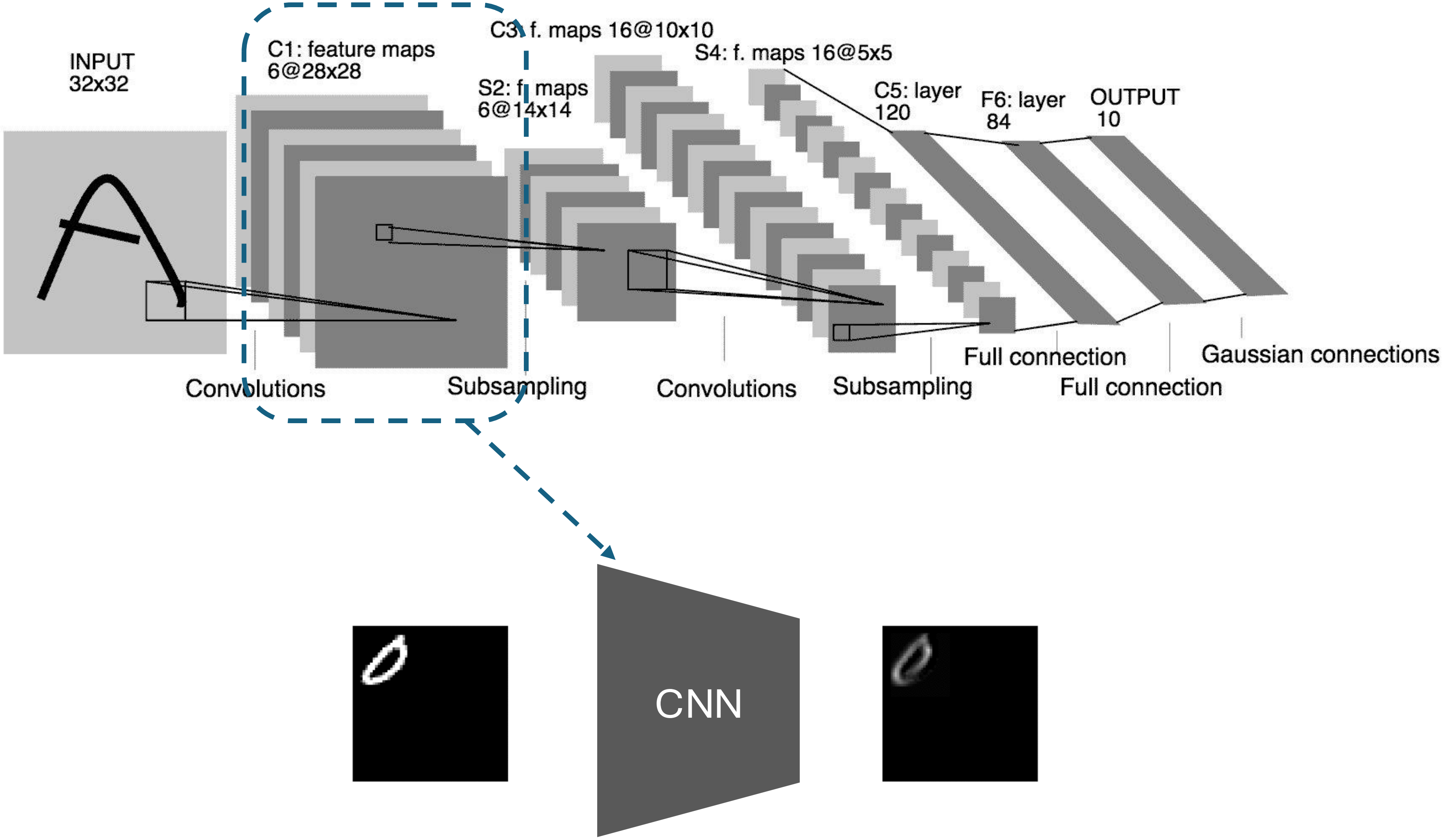
$$\mathbf{R}f(\boldsymbol{x}) = f(\mathbf{R}\boldsymbol{x})$$

Background: Equivariance





Background: Equivariance



Background: Equivariance

In molecular modelling we are interested in equivariance w.r.t.:

- Translations 3D
- Rotations 3D (possibly reflections)
- Permutations

Background: Equivariance

In molecular modelling we are interested in equivariance w.r.t.:

- Translations 3D
- Rotations 3D (possibly reflections)
- Permutations

E(3) Equivariance

Background: Equivariance

In molecular modelling we are interested in equivariance w.r.t.:

- Translations 3D
- Rotations 3D (possibly reflections)

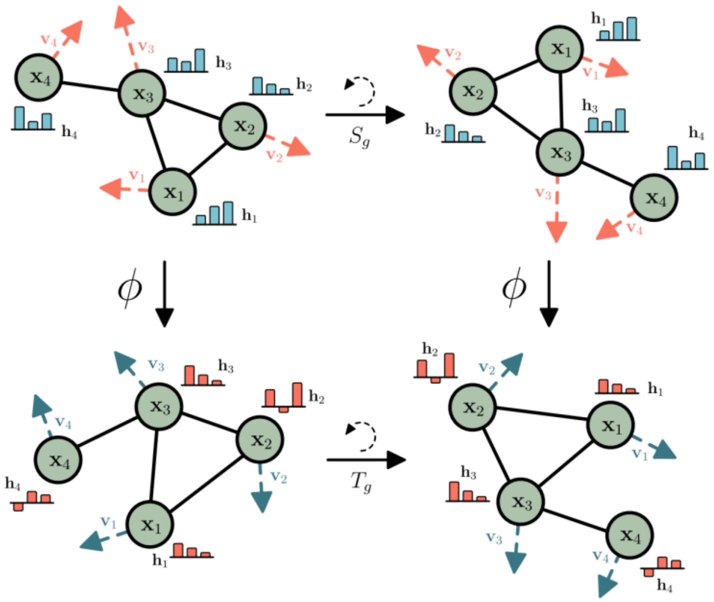
E(3) Equivariance

- Permutations -> Graph Neural Networks / Transformers

Background: Equivariance

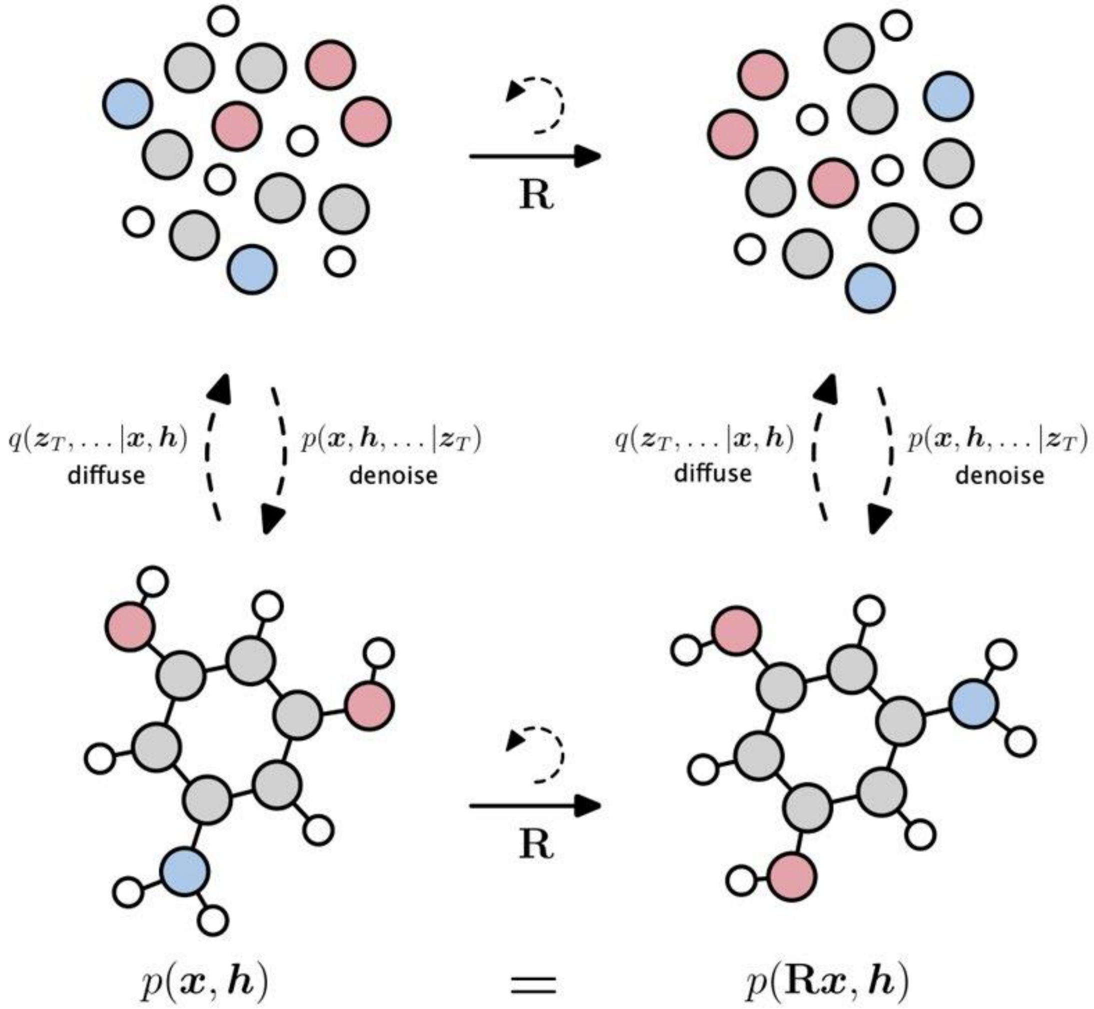
E(n) Equivariant Graph Neural Networks

Victor Garcia Satorras [1] Emiel Hoogeboom [1] Max Welling [1]



EDM: Equivariant Diffusion Models

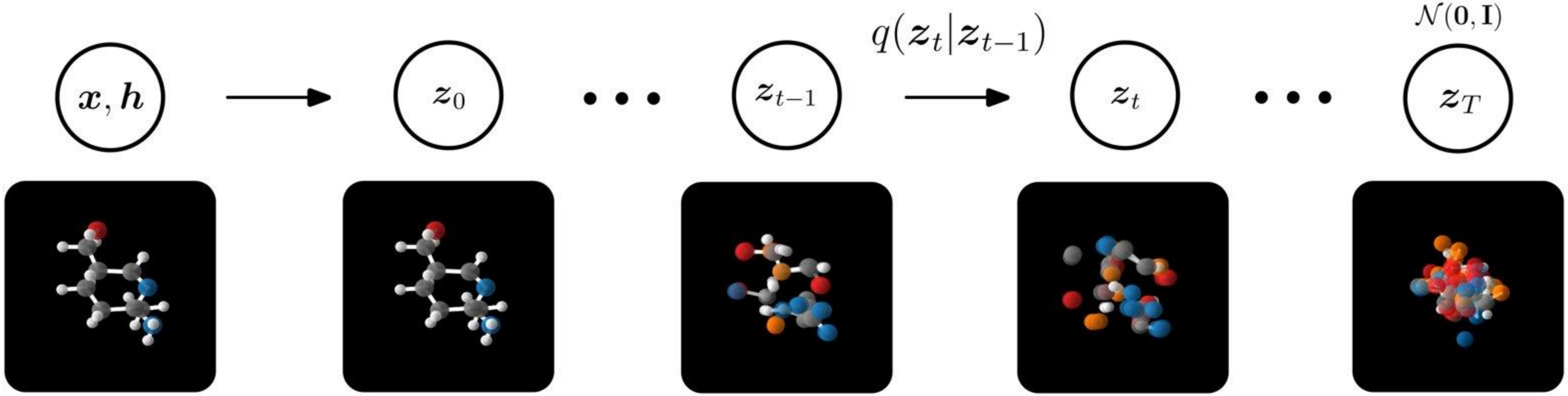
- Diffusion process to destroy info
- Learn denoising process to generate
- Handles continuous and discrete data



Diffusion Process

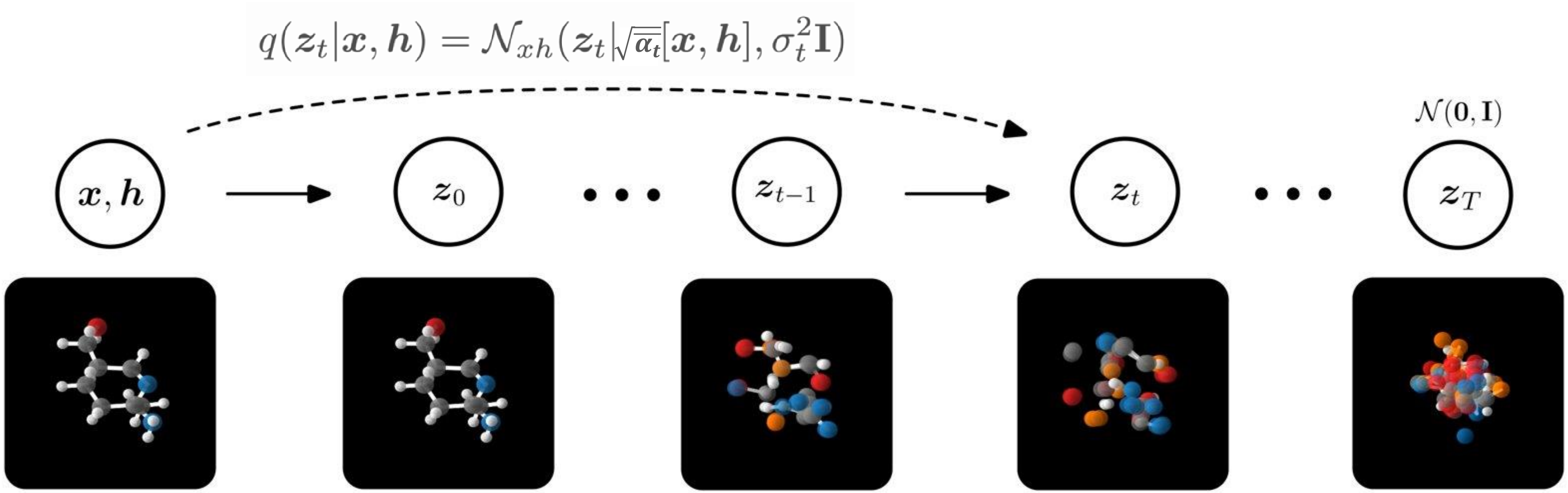
- Adds Gaussian noise over time steps t = 0, ..., T

- Is equivariant to rotations and translations



Diffusion Process

- Jump to time step “t”

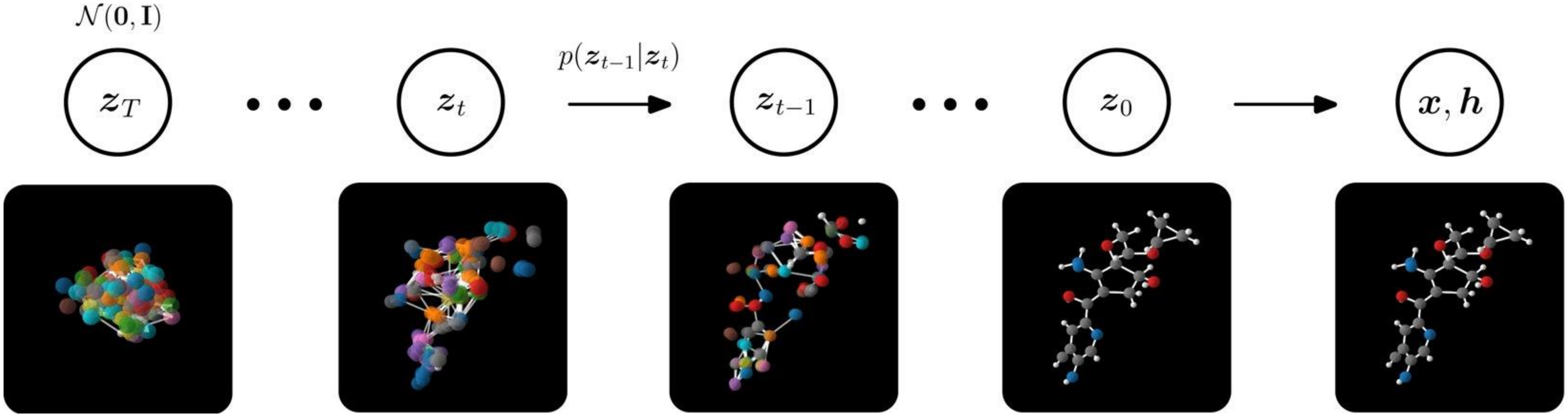


To generate molecules

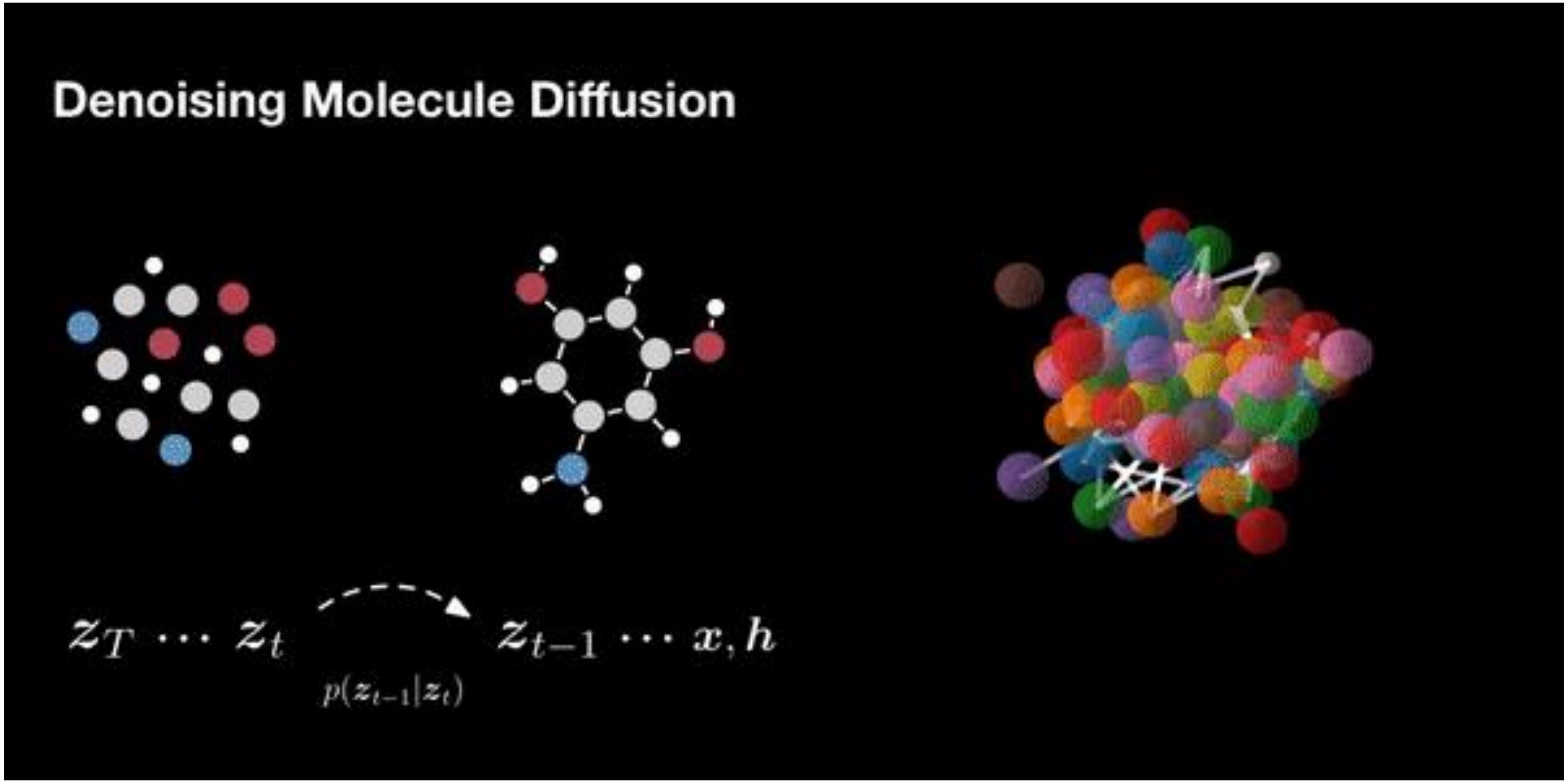
Learn the *reverse* denoising process

Learnable Denoising Process

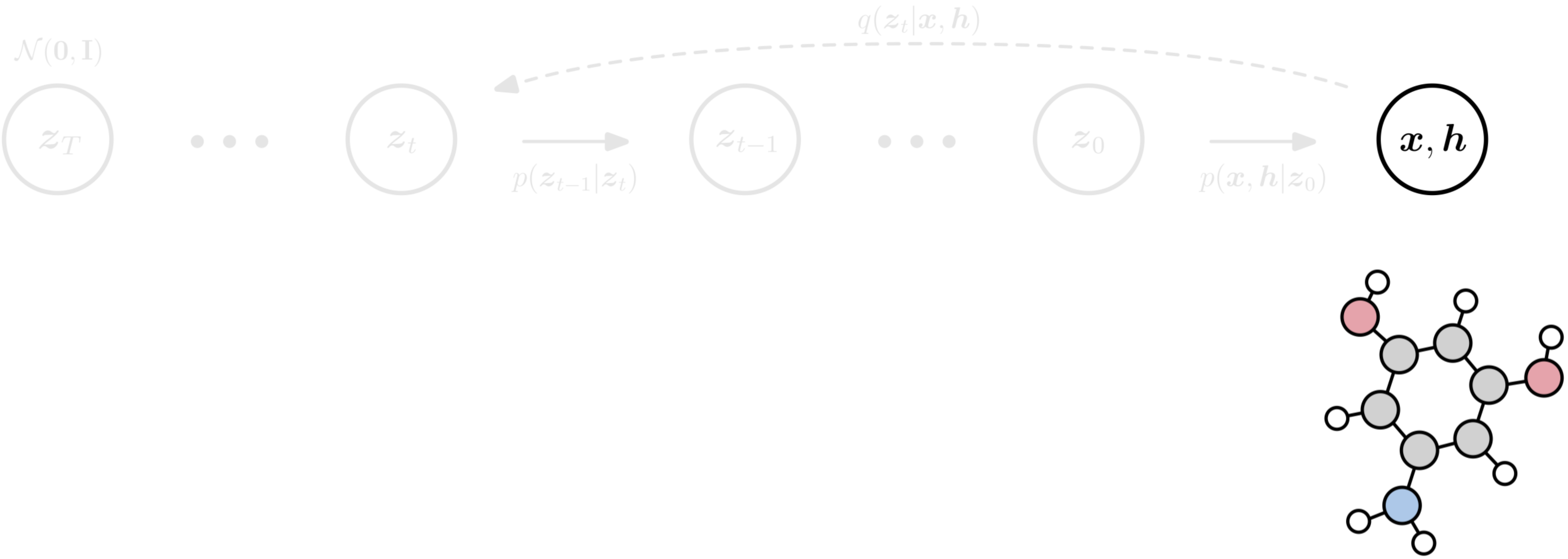
- Denoise with distributions. $p\left(\boldsymbol{z}_{t-1} \mid \boldsymbol{z}_t\right) \approx \mathcal{N}\left(\boldsymbol{z}_{t-1} \mid \hat{\boldsymbol{\mu}}_{t \rightarrow t-1}\left(\boldsymbol{z}_t\right), \sigma_{t \rightarrow t-1}^2 \mathbf{I}\right)$
- Chosen to have same form as the true denoising process.



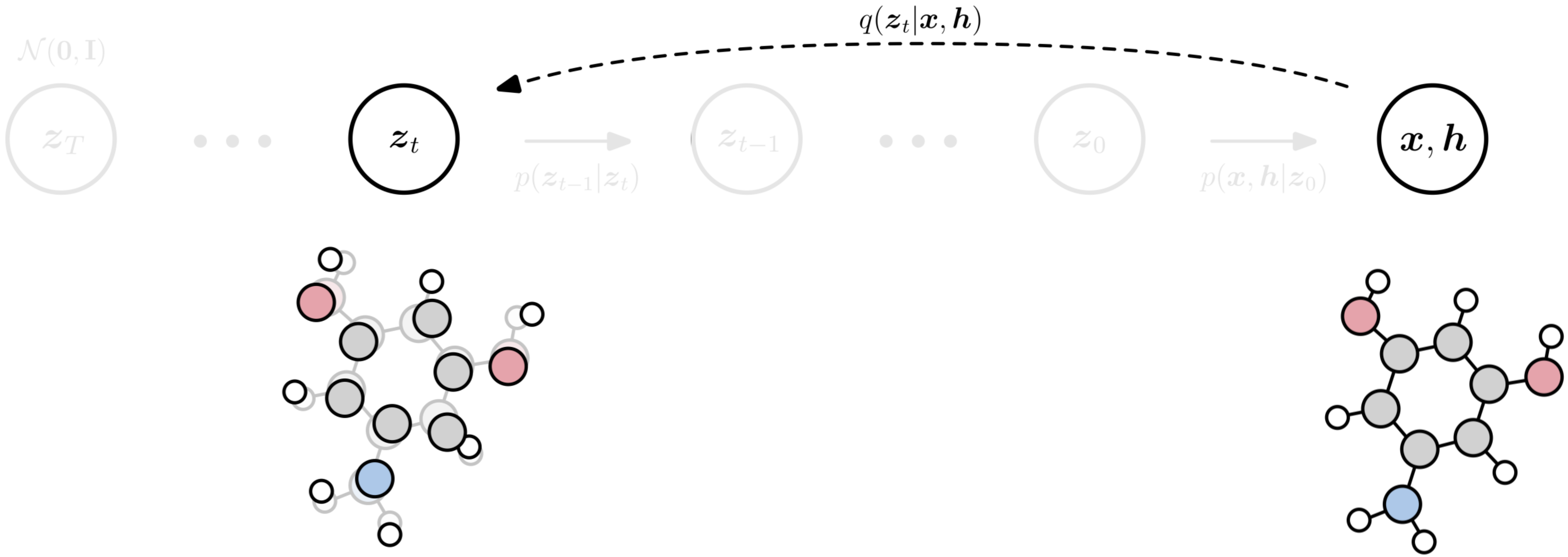
Equivariant Diffusion Models



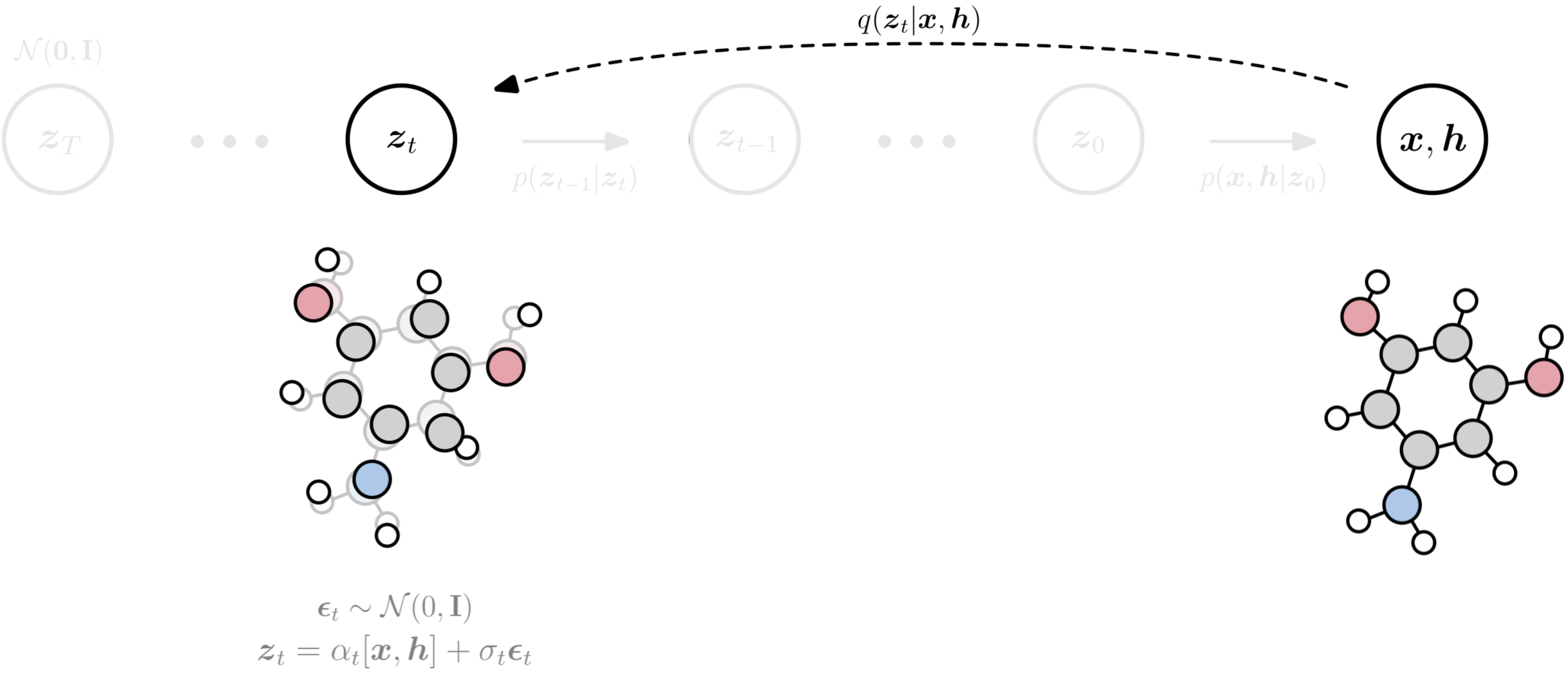
Optimization



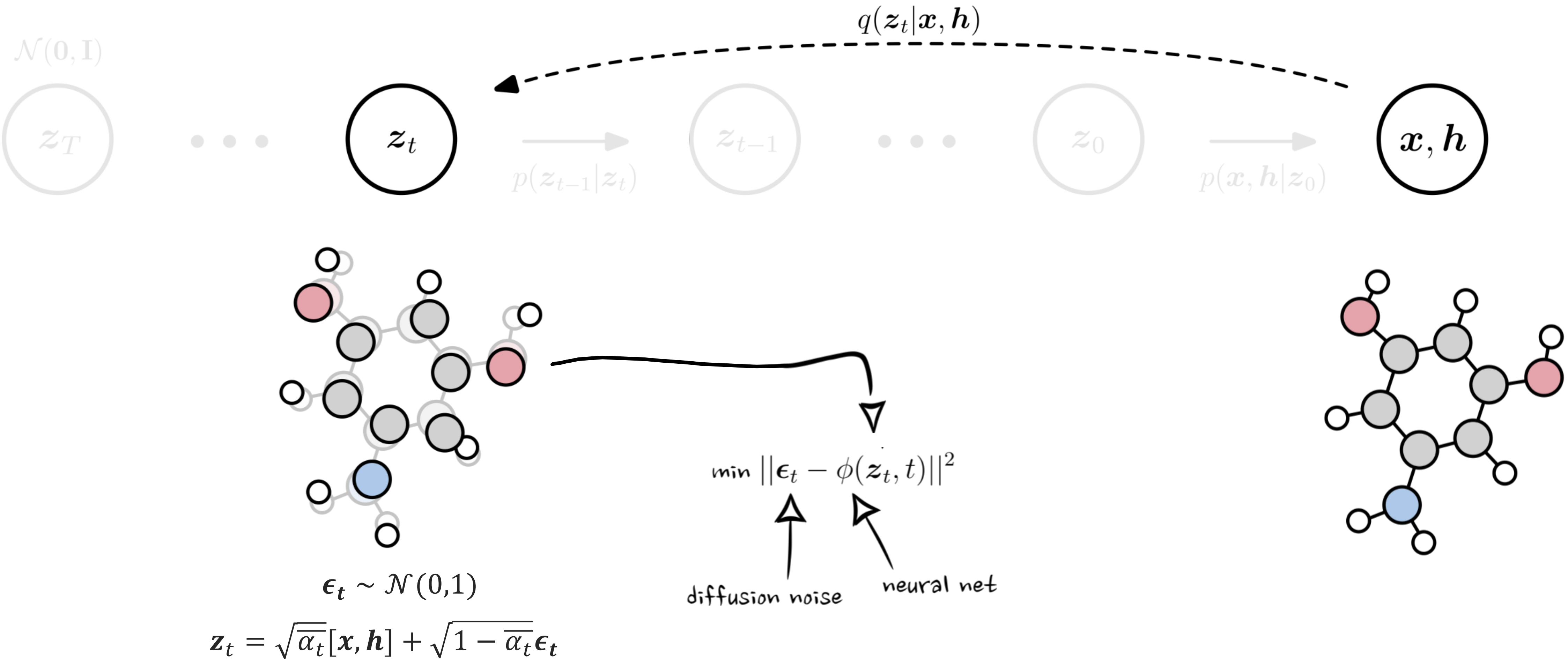
Optimization



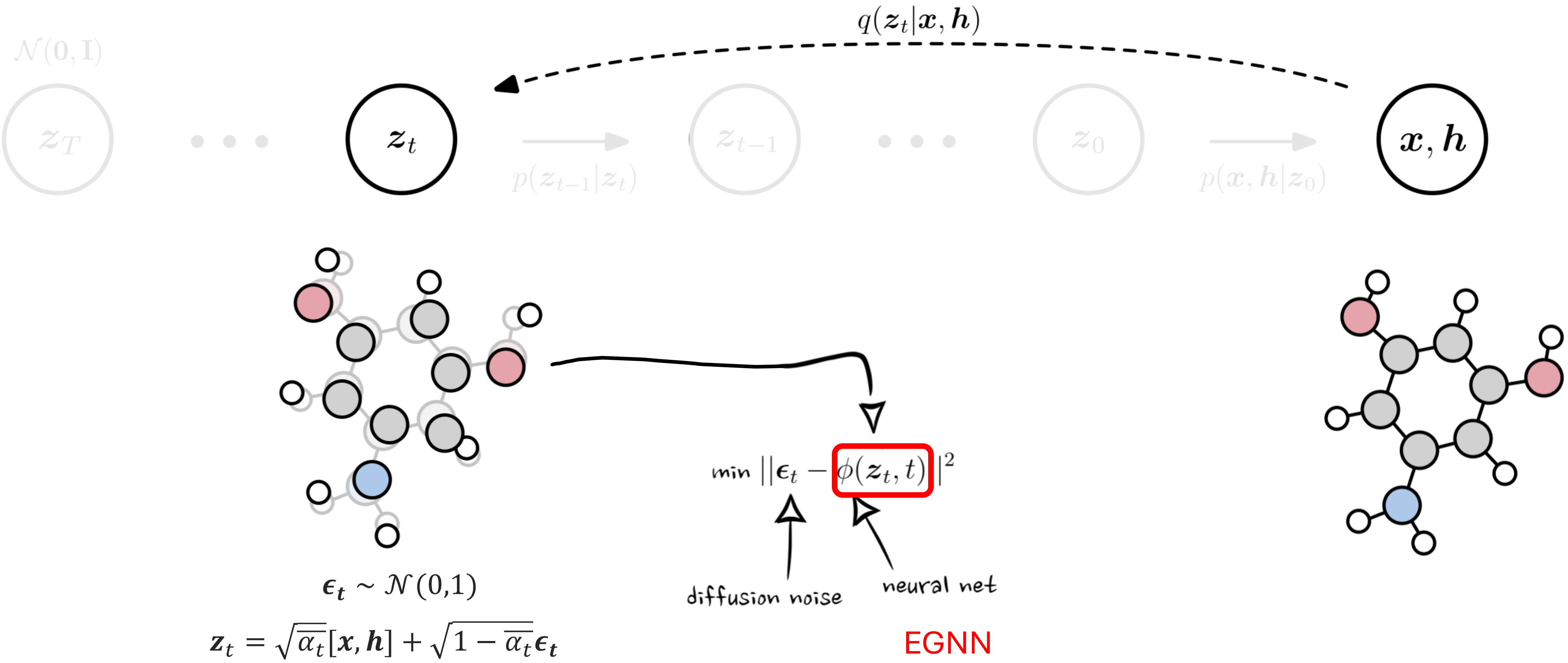
Optimization



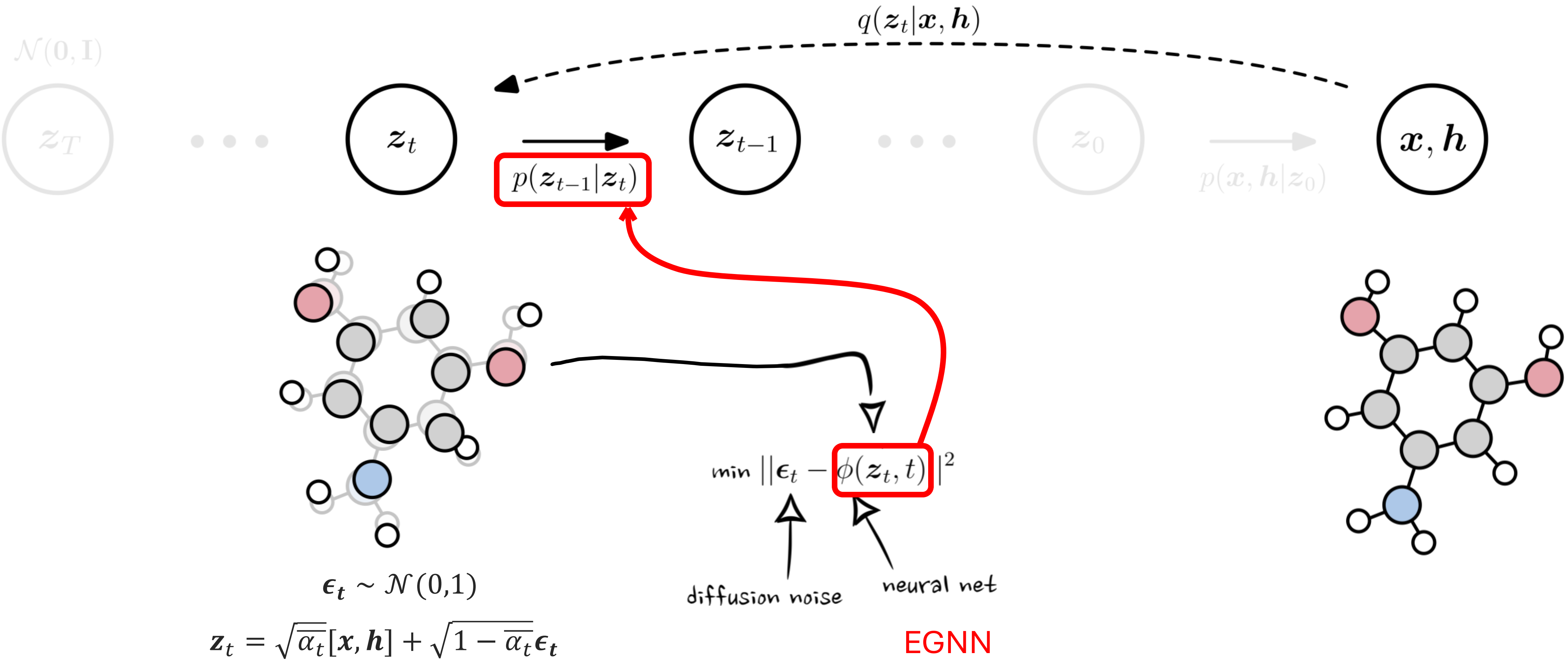
Optimization



Optimization



Optimization

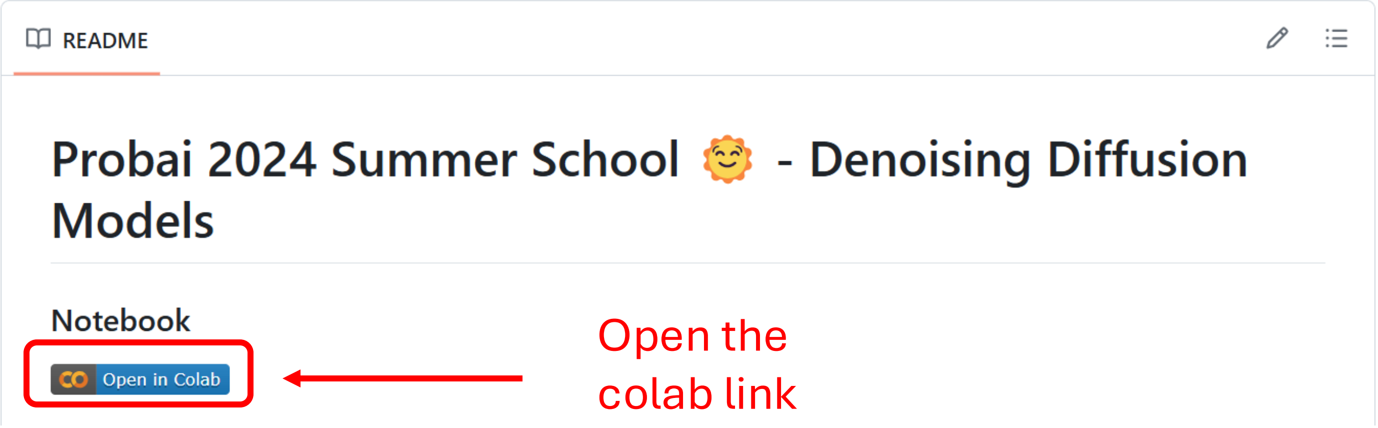


Outline

1. Introduction to Denoising Diffusion Models
2. Equivariant Diffusion Models for Molecule Generation in 3D
3. **Workshop: Code & Practice**

Workshop: Code & Practice

1. Open github repository: https://github.com/vgsatorras/probai24
2. Open the Collab:



3. Copy the colab file to your personal drive: