block b6

block b6

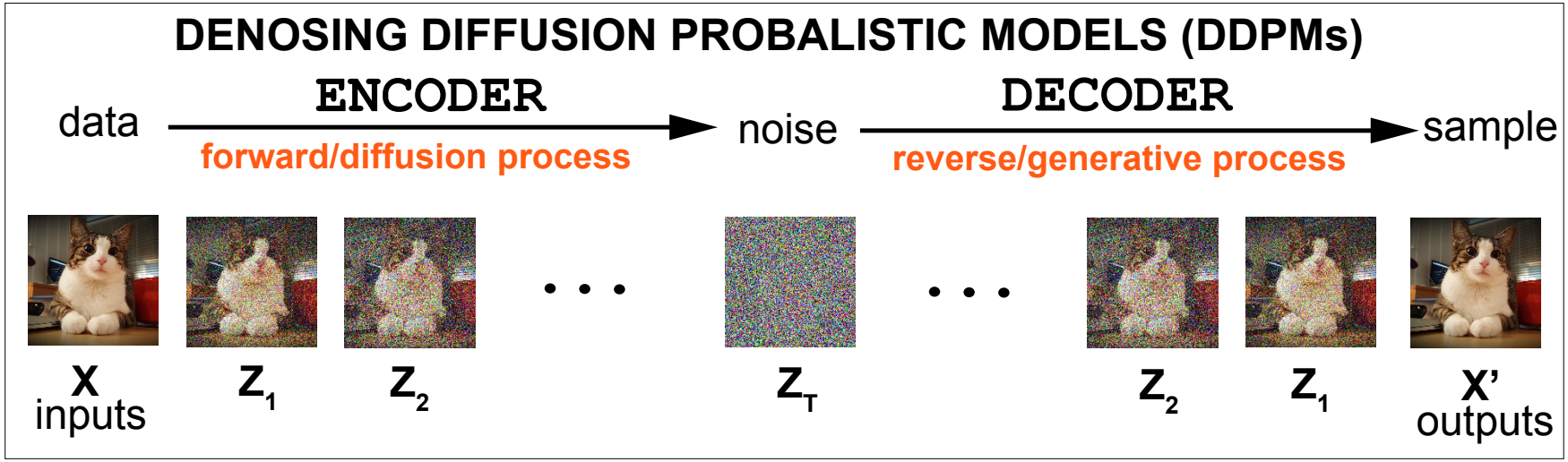
Diffusion Models, Flow Matching Models

block b6

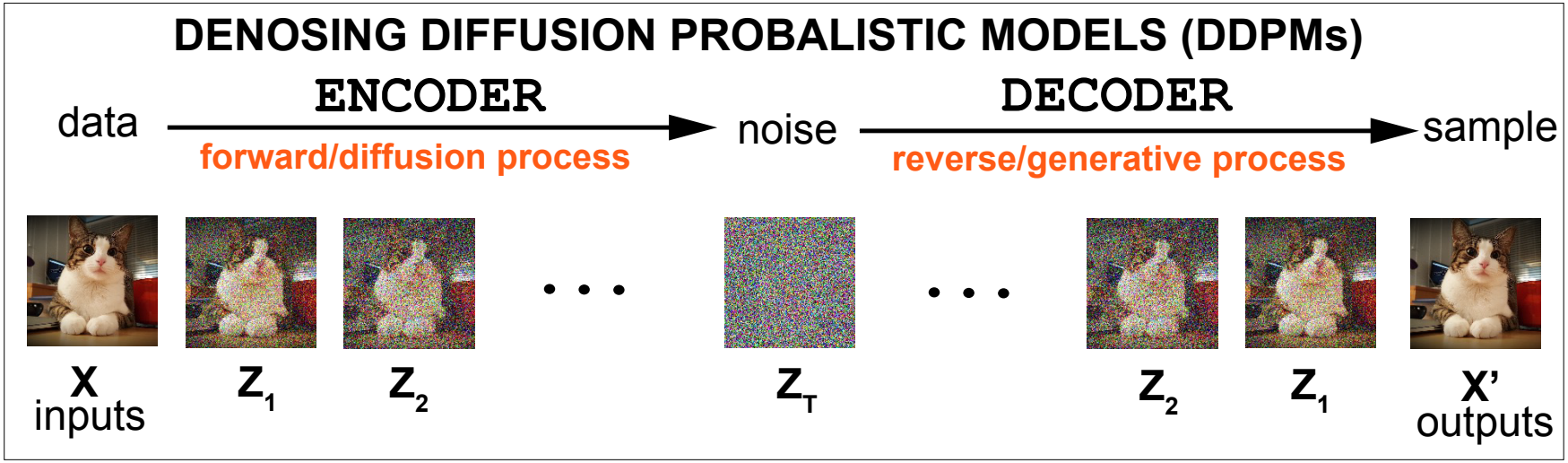
Diffusion Models, Flow Matching Models

Protein Design, RNA seq & structure generation

Denoising Diffusion Models

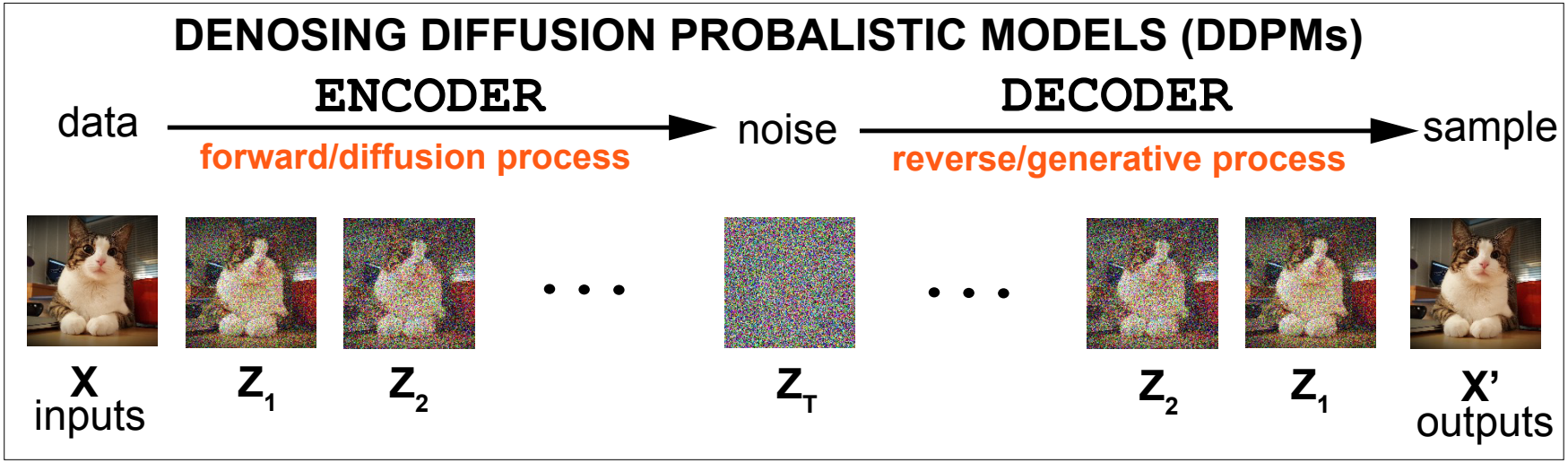


Denoising Diffusion Models



Encoder / Decoder

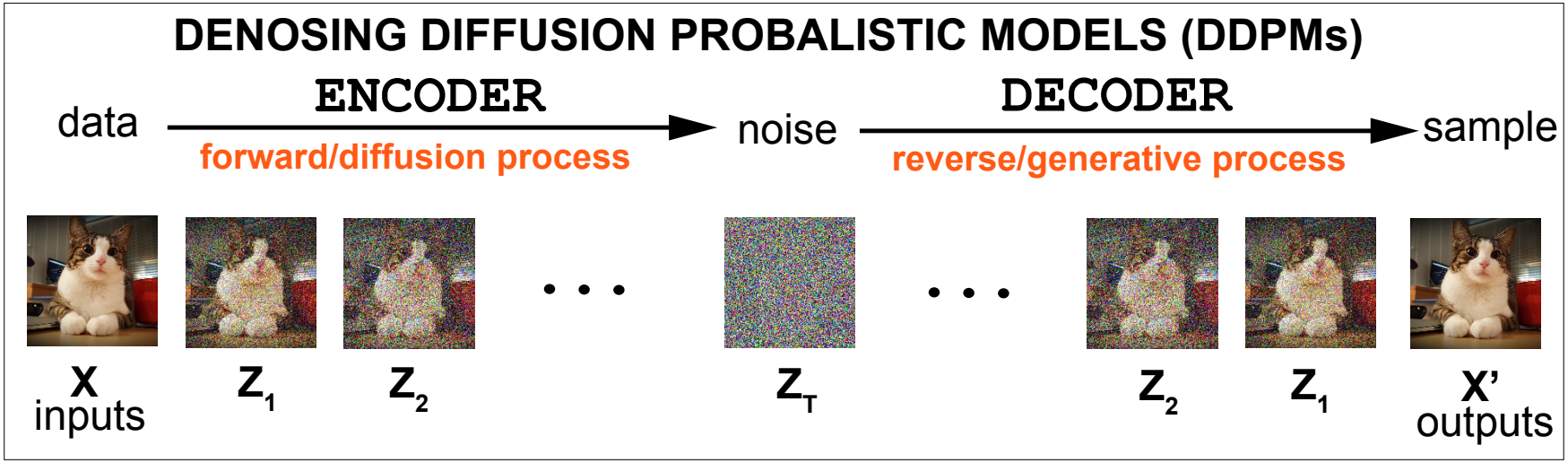
Denoising Diffusion Models



Encoder / Decoder

Latent variable

Denoising Diffusion Models

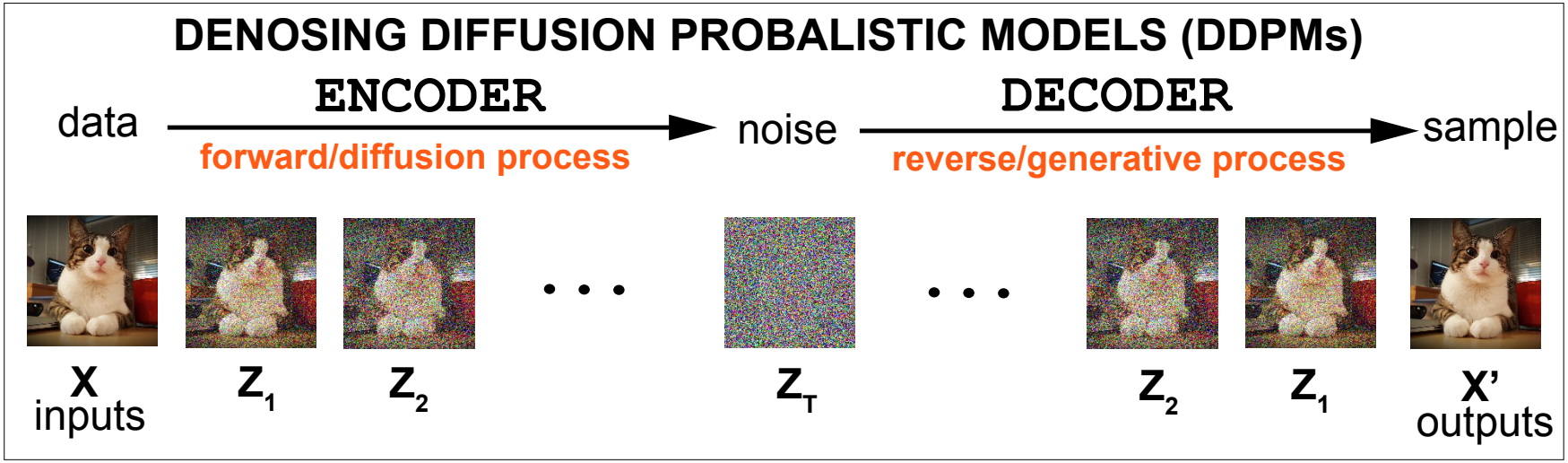


Encoder / Decoder

Latent variable

Probabilitic / Generative

Denoising Diffusion Models



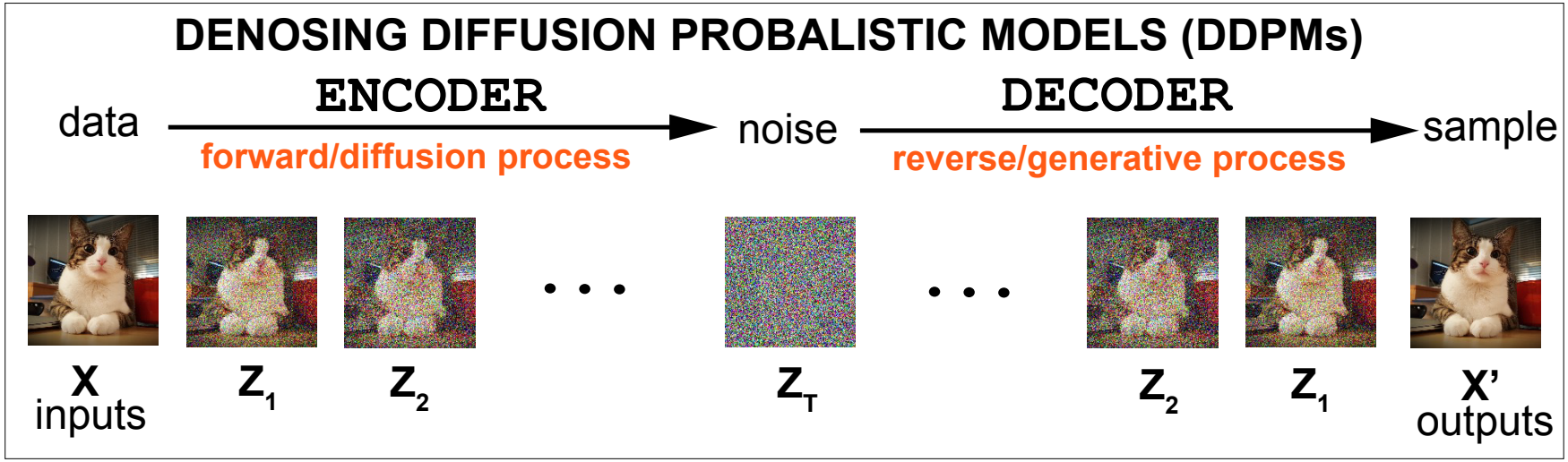
Encoder / Decoder

Latent variable

Probabilitic / Generative

Use a variational distribution

Denoising Diffusion Models



Encoder / Decoder

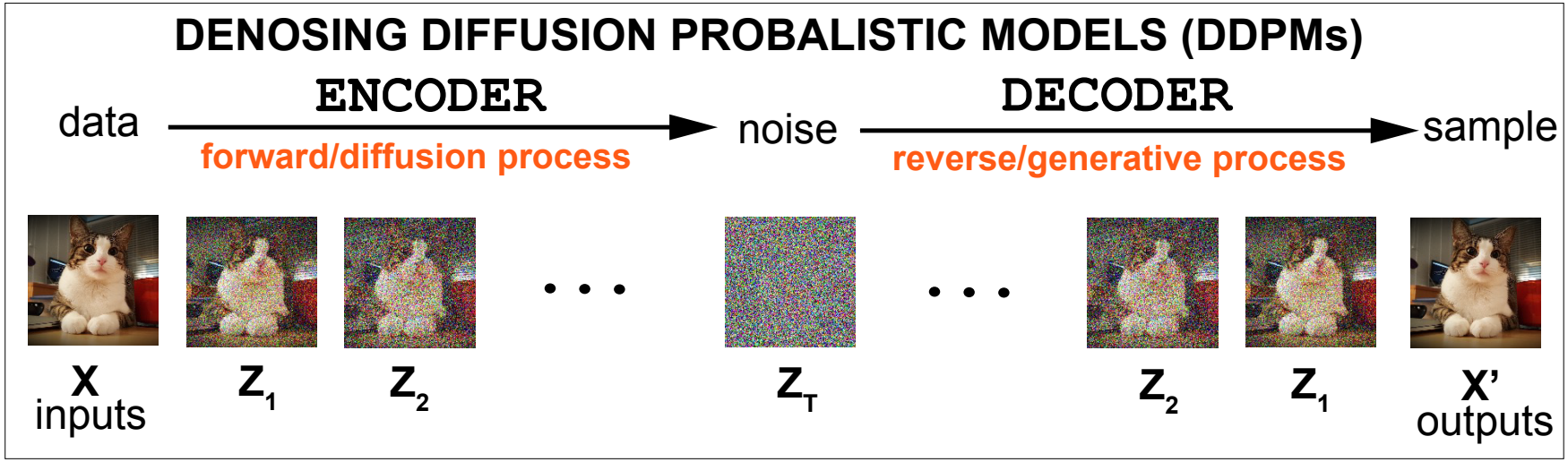
Latent variable

Probabilitic / Generative

Use a variational distribution

Unsupervised

Denoising Diffusion Models



Encoder / Decoder

Latent variable

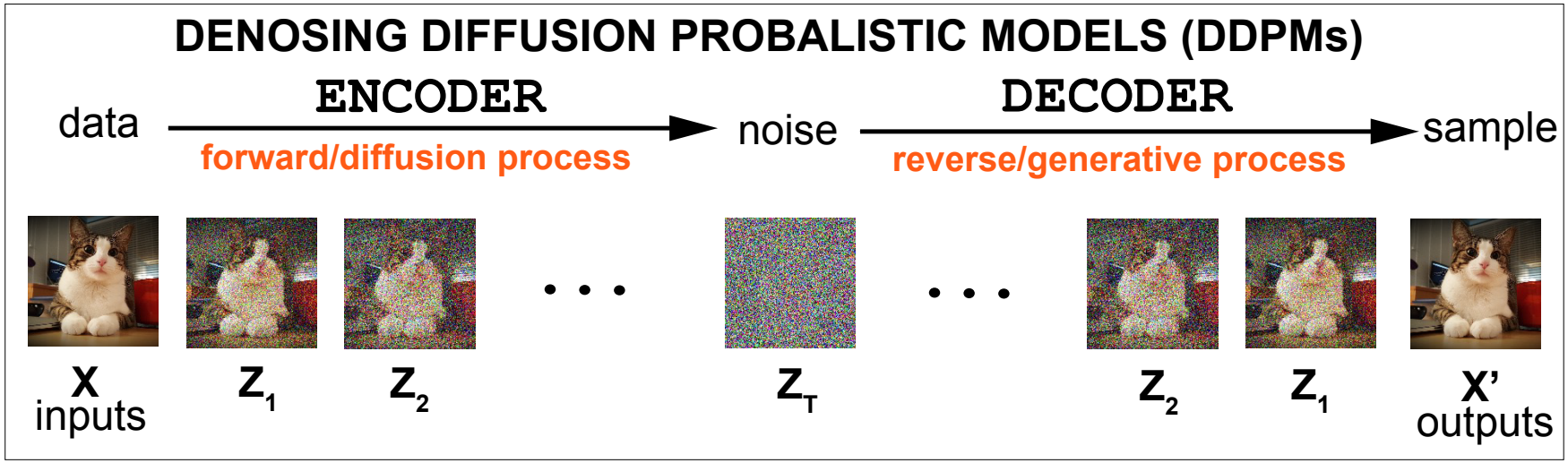
Probabilitic / Generative

Use a variational distribution

Unsupervised

Approximate P_{data} using a lower bound

Denoising Diffusion Models



Encoder / Decoder

Latent variable

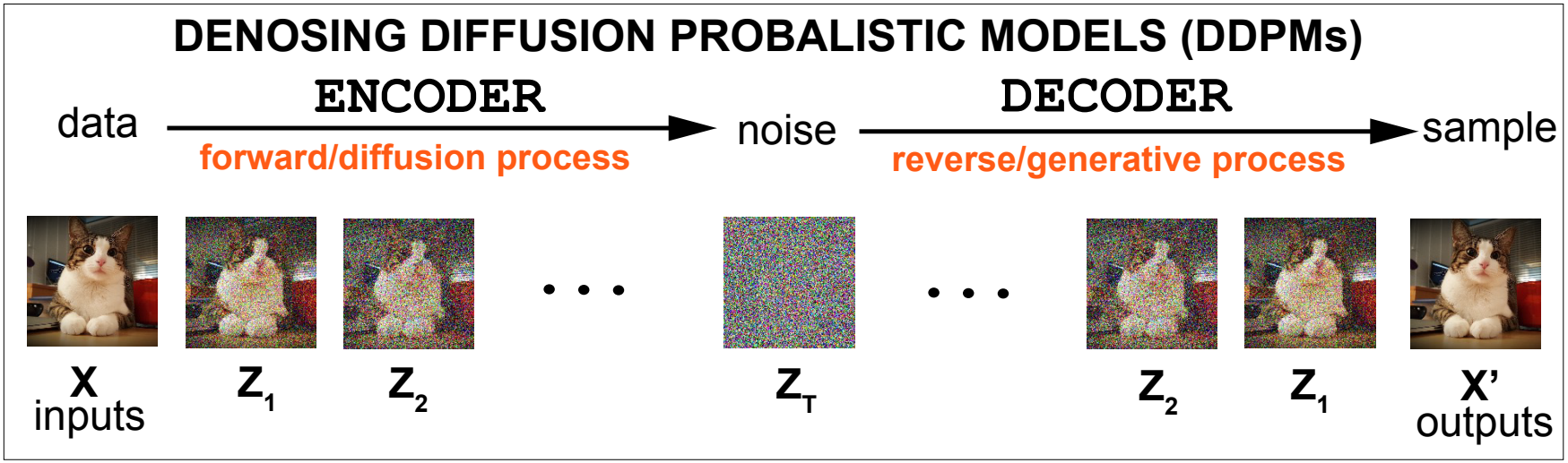
Probabilitic / Generative

Use a variational distribution

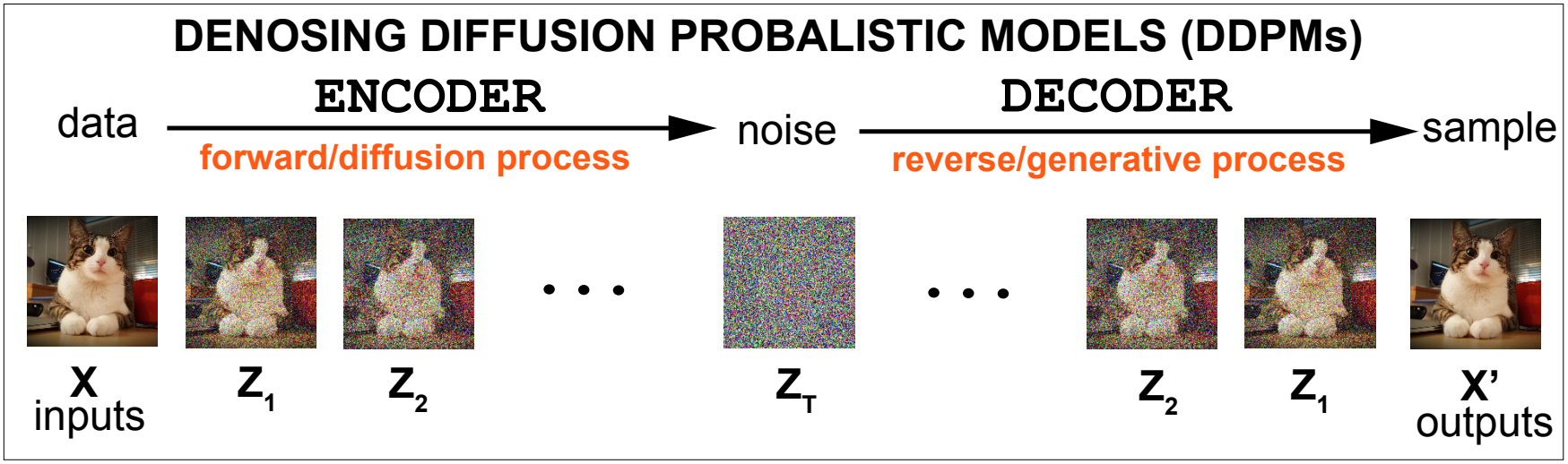
Unsupervised

Approximate P_{data} using a lower bound

they look a lot like VAEs

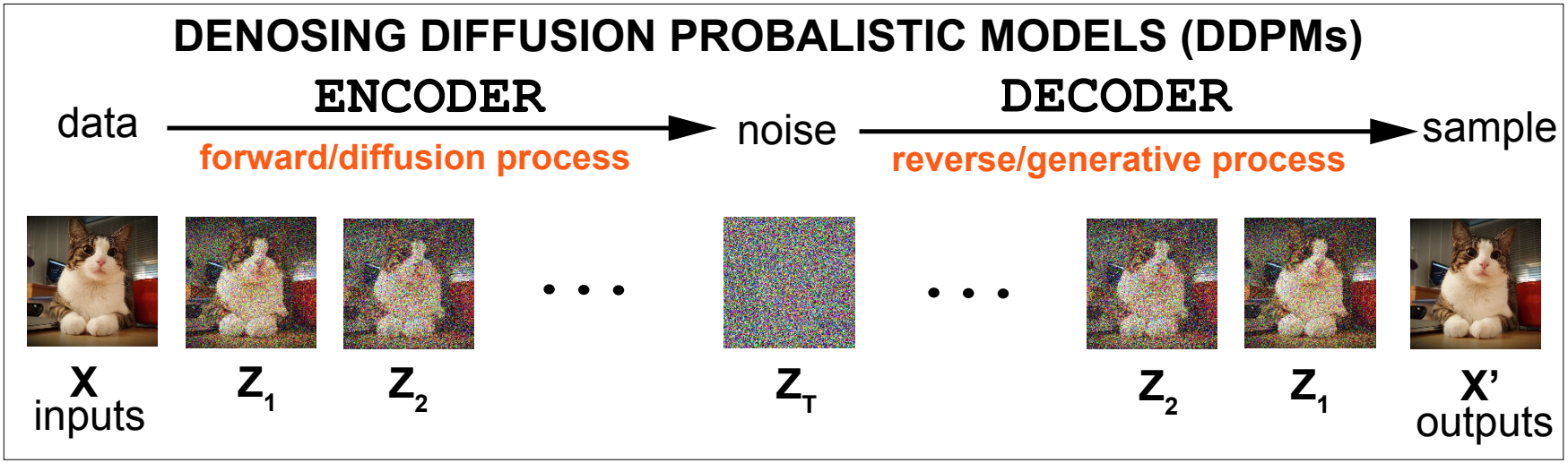


but they are not VAEs



but they are not VAEs

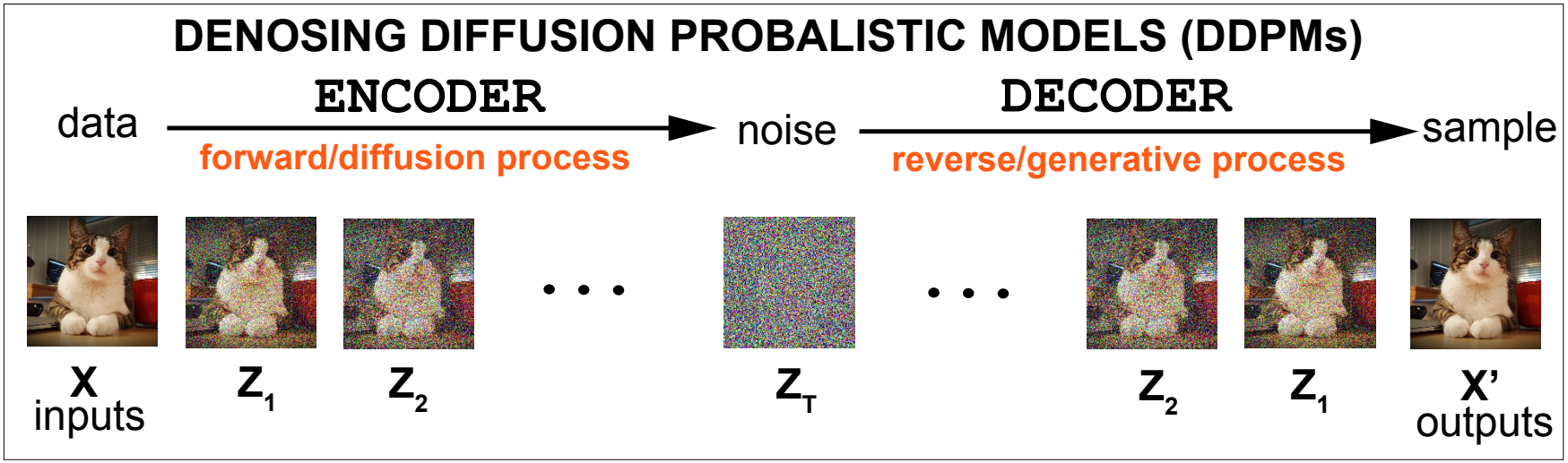
There is a time series of latent variables



but they are not VAEs

There is a time series of latent variables

$$Z_1, Z_2, \cdots, Z_T$$

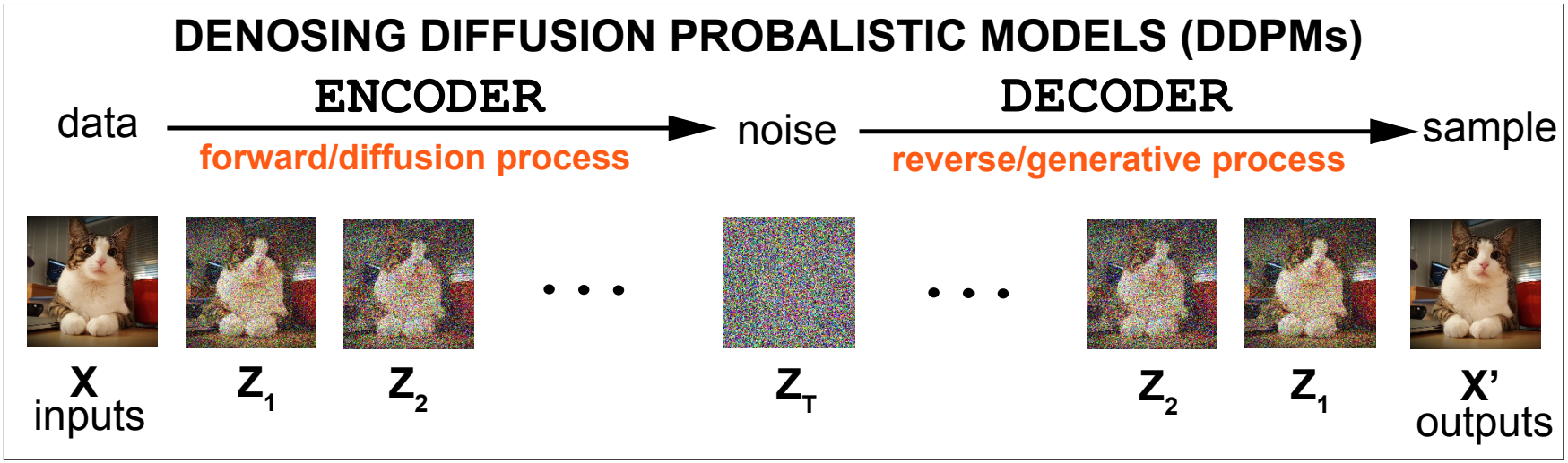


but they are not VAEs

There is a time series of latent variables

$$Z_1, Z_2, \cdots, Z_T$$

all of same dimension



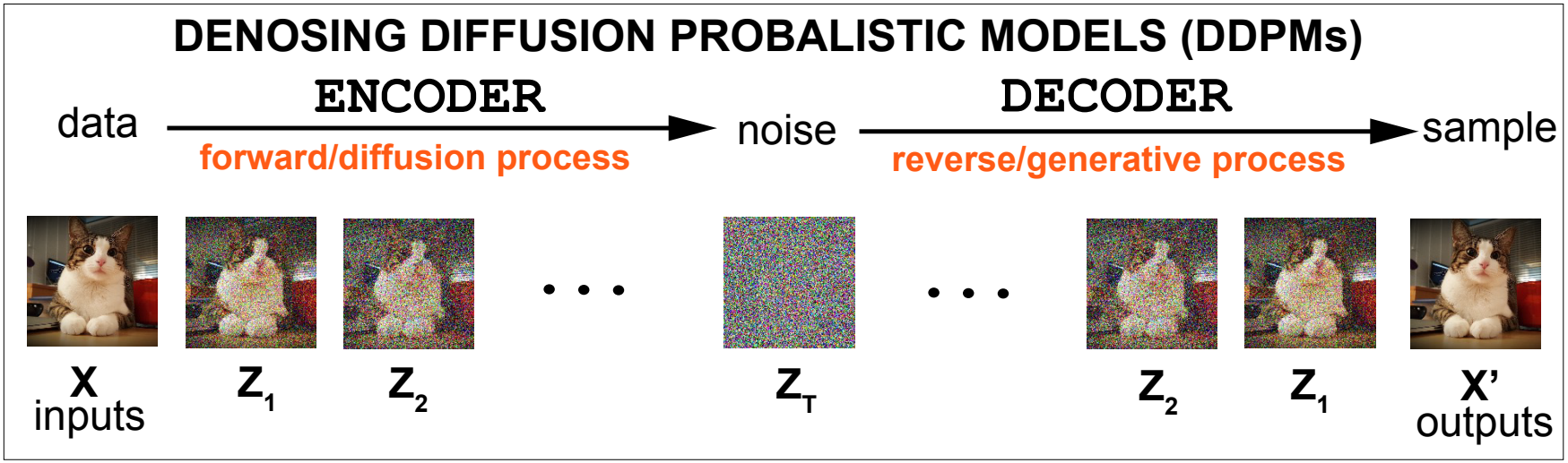
but they are not VAEs

There is a time series of latent variables

$$Z_1, Z_2, \cdots, Z_T$$

all of same dimension

The encoder is completely specified



but they are not VAEs

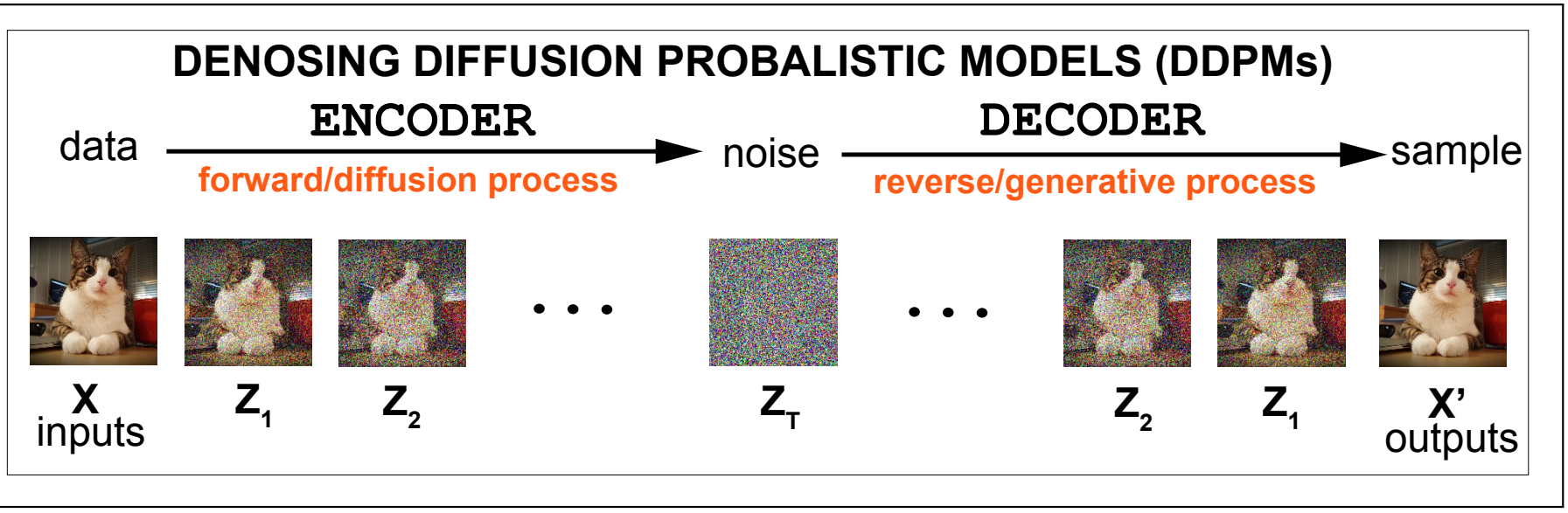
There is a time series of latent variables

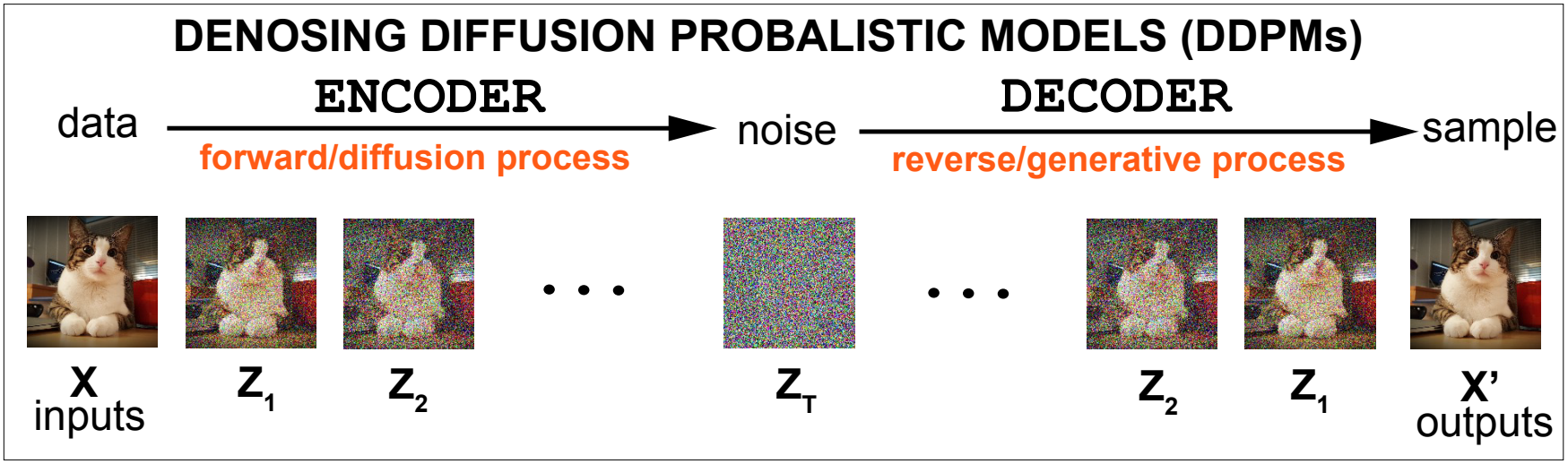
$$Z_1, Z_2, \cdots, Z_T$$

all of same dimension

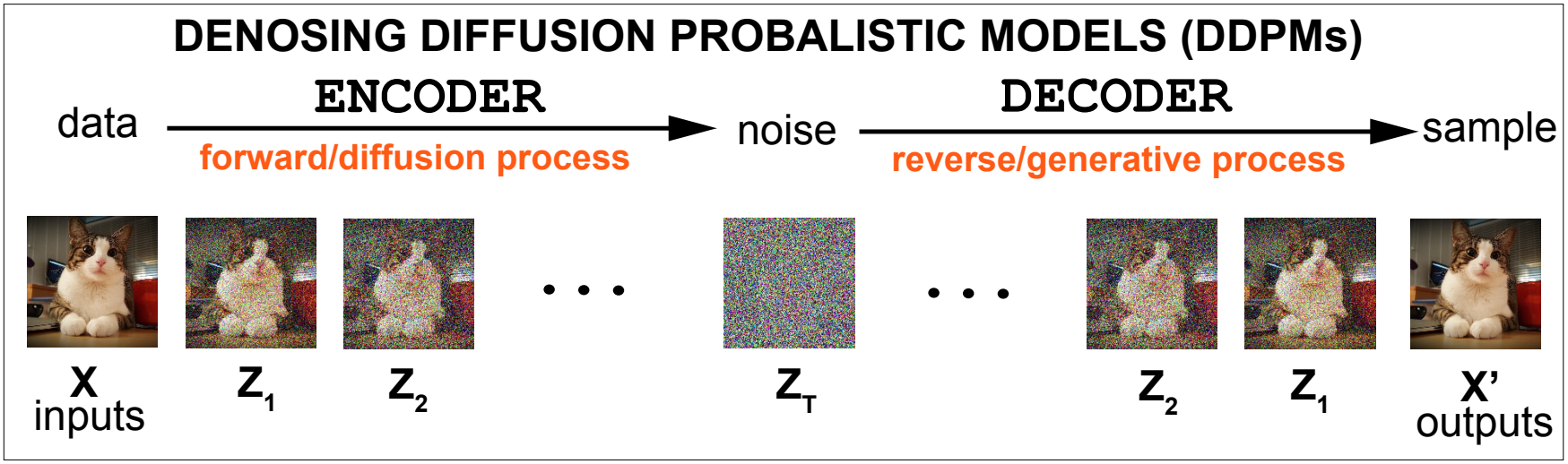
The encoder is completely specified

All parameters are in the Decoder


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
inputs
Z_1
Z_2
Z_T
Z_2
Z_1
X'
outputs

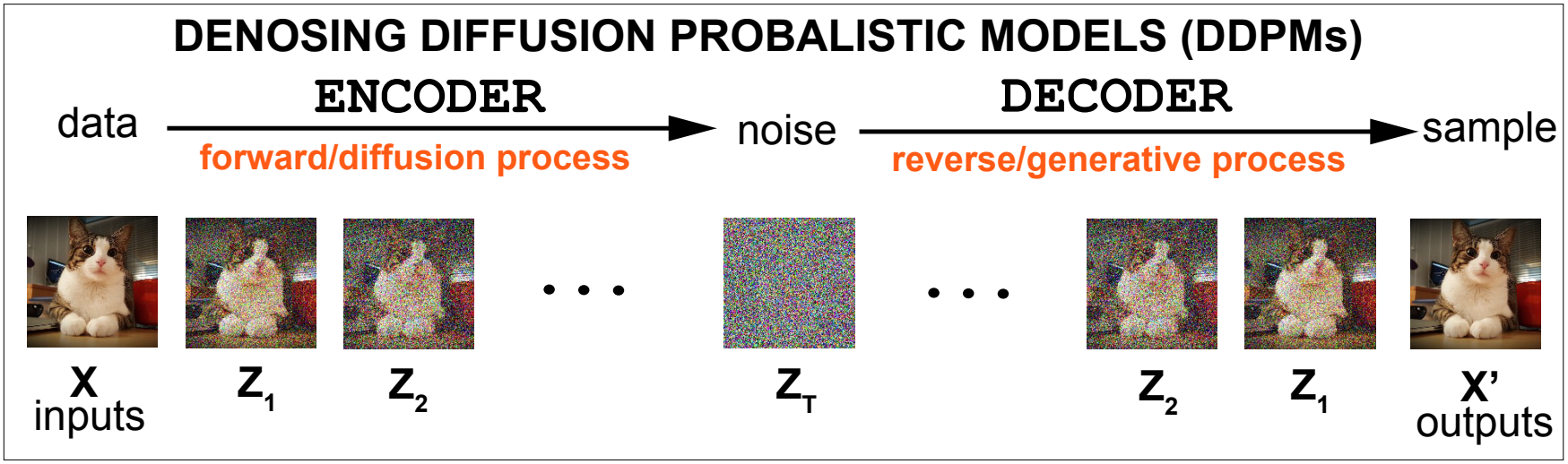


Encoder = diffusion



Encoder = diffusion

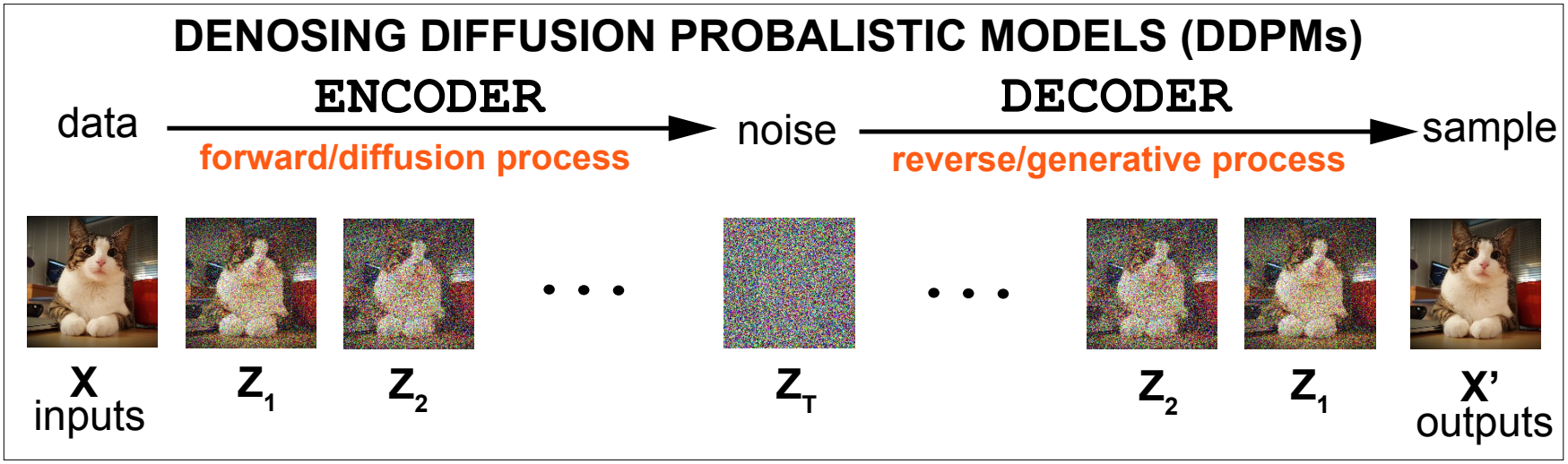
Start with X



Encoder = diffusion

Start with X

add some noise $Z_1 = X +$ noise

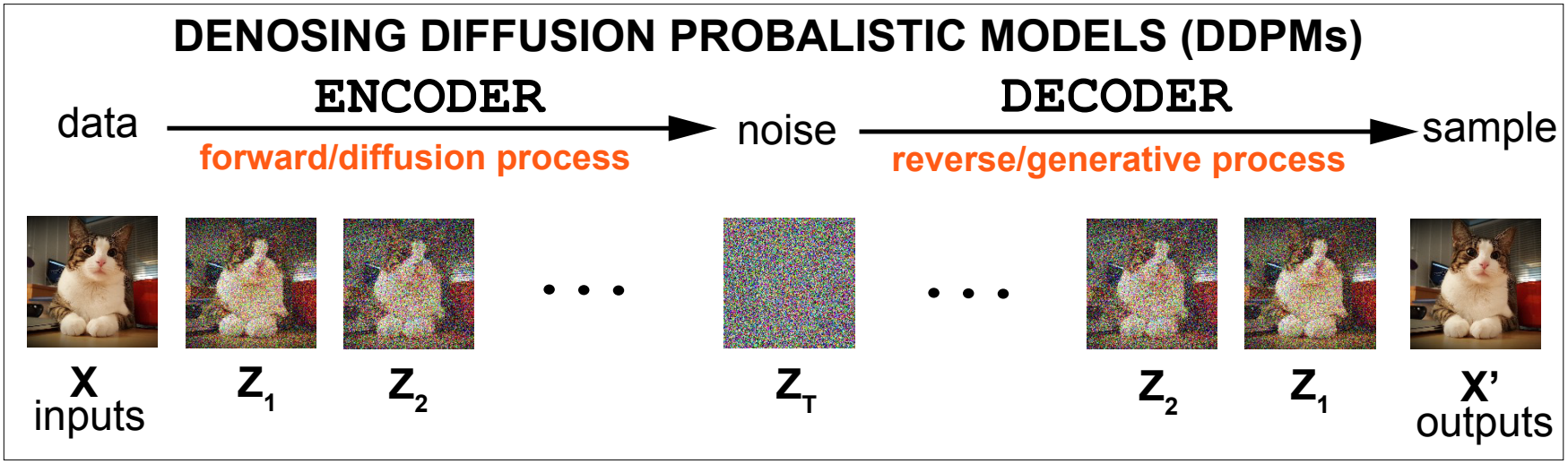


Encoder = diffusion

Start with X

add some noise $Z_1 = X + \text{noise}$

add more noise $Z_2 = Z_1 + \text{noise}$



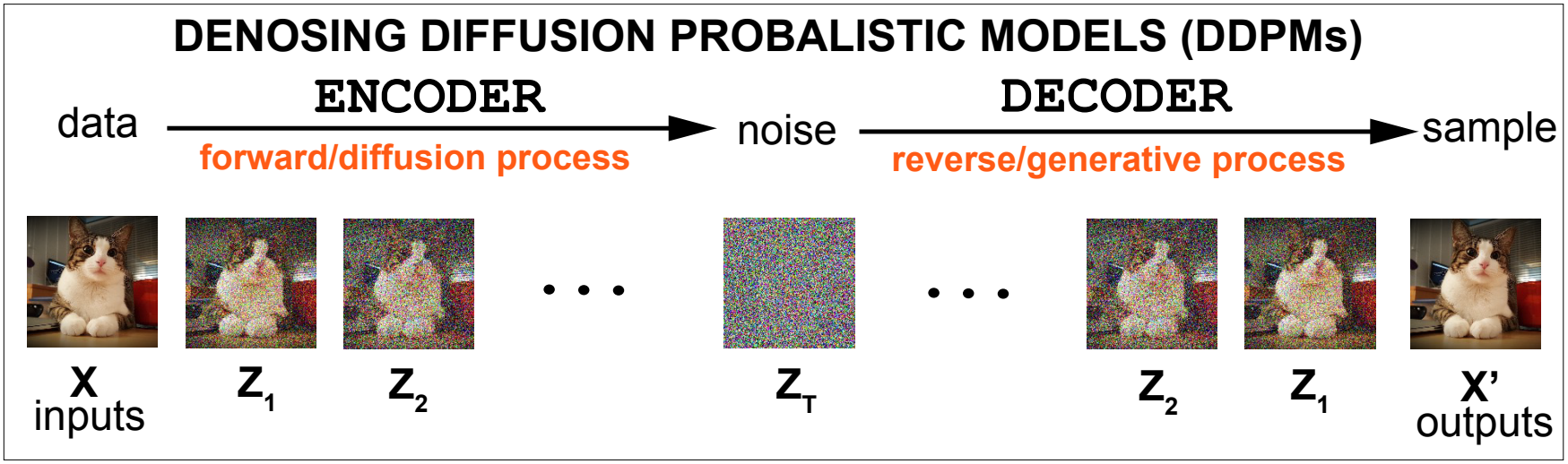
Encoder = diffusion

Start with X

add some noise $Z_1 = X + \text{noise}$

add more noise $Z_2 = Z_1 + \text{noise}$

.. add more noise $Z_T = Z_{T-1} + \text{noise}$



Encoder = diffusion

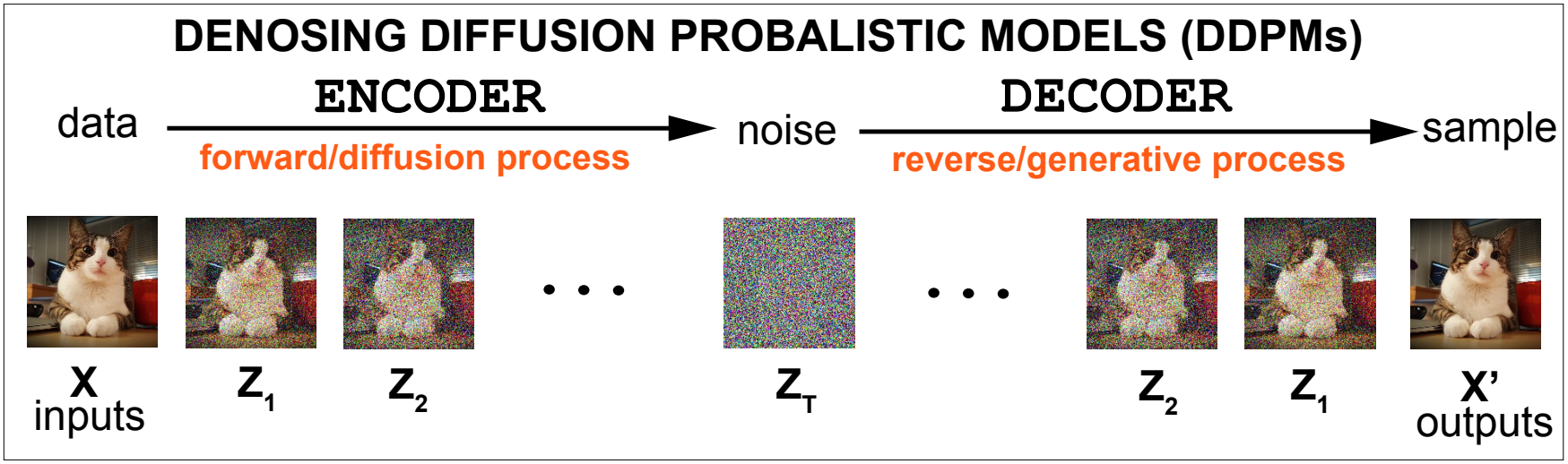
Start with X

add some noise $Z_1 = X + \text{noise}$

add more noise $Z_2 = Z_1 + \text{noise}$

.. add more noise $Z_T = Z_{T-1} + \text{noise}$

$P(Z_T|X) = N(0, I)$



Encoder = diffusion

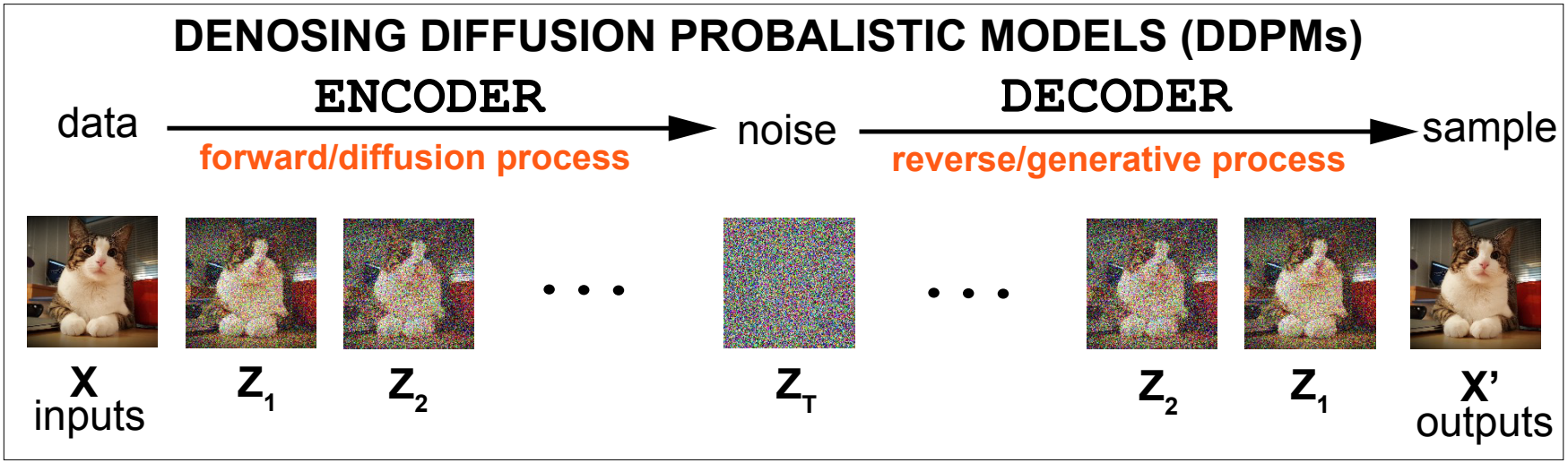
Start with X

add some noise $Z_1 = X + \text{noise}$

add more noise $Z_2 = Z_1 + \text{noise}$

.. add more noise $Z_T = Z_{T-1} + \text{noise}$

$$P(Z_T|X) = N(0, I) \; P(Z_T) = N(0, I)$$



Encoder = diffusion

Start with X

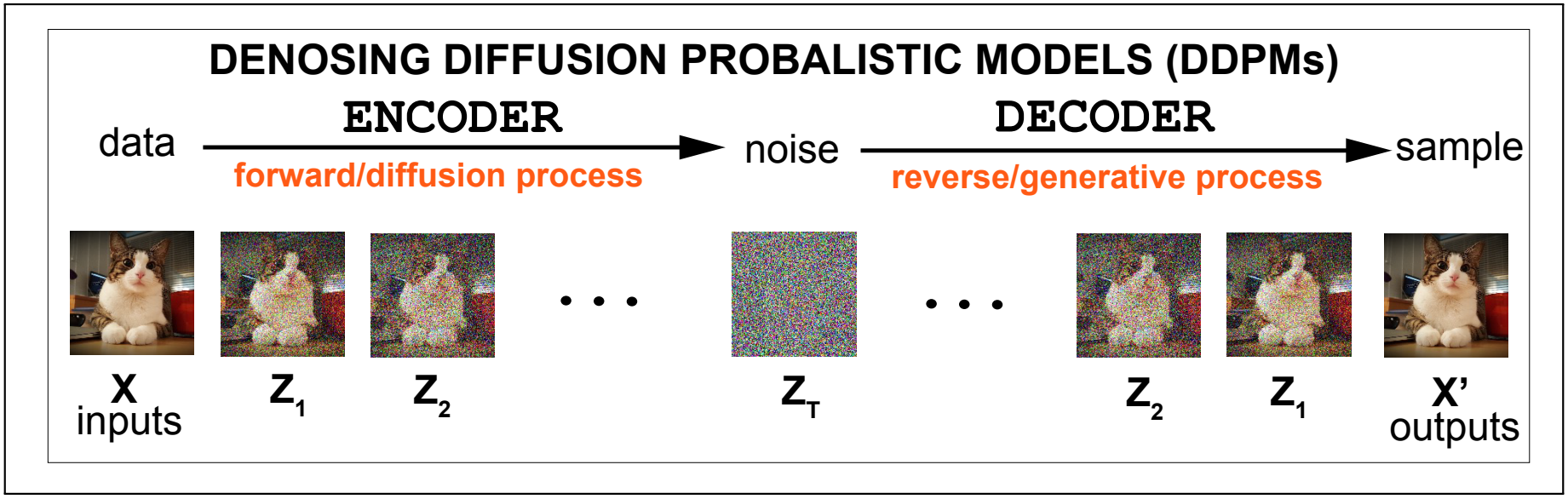
add some noise $Z_1 = X + \text{noise}$

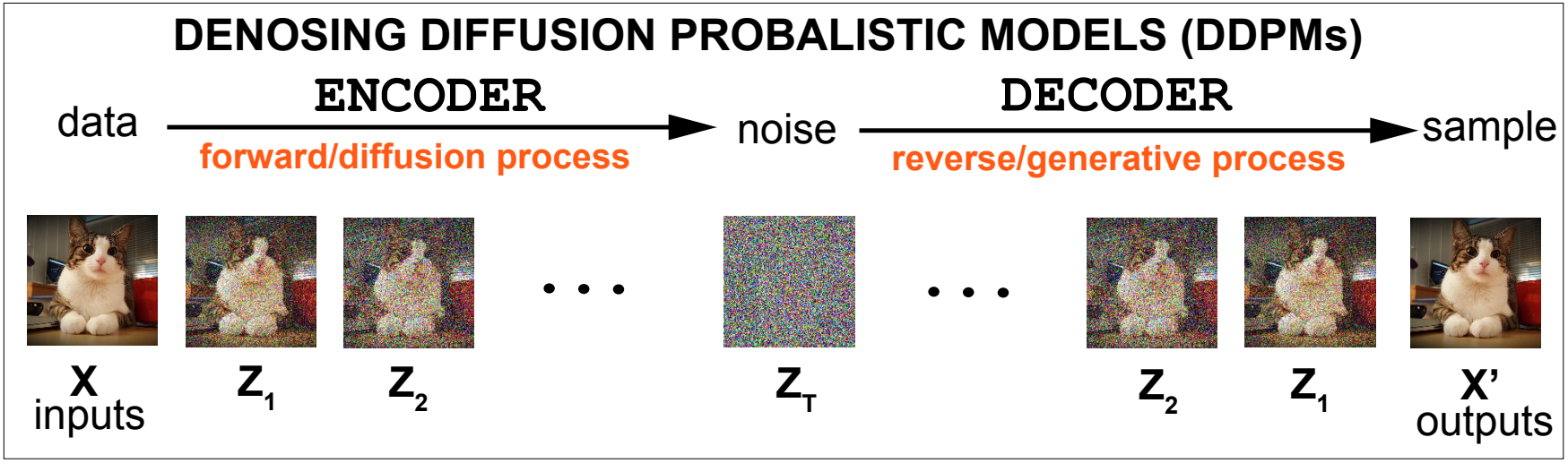
add more noise $Z_2 = Z_1 + \text{noise}$

.. add more noise $Z_T = Z_{T-1} + \text{noise}$

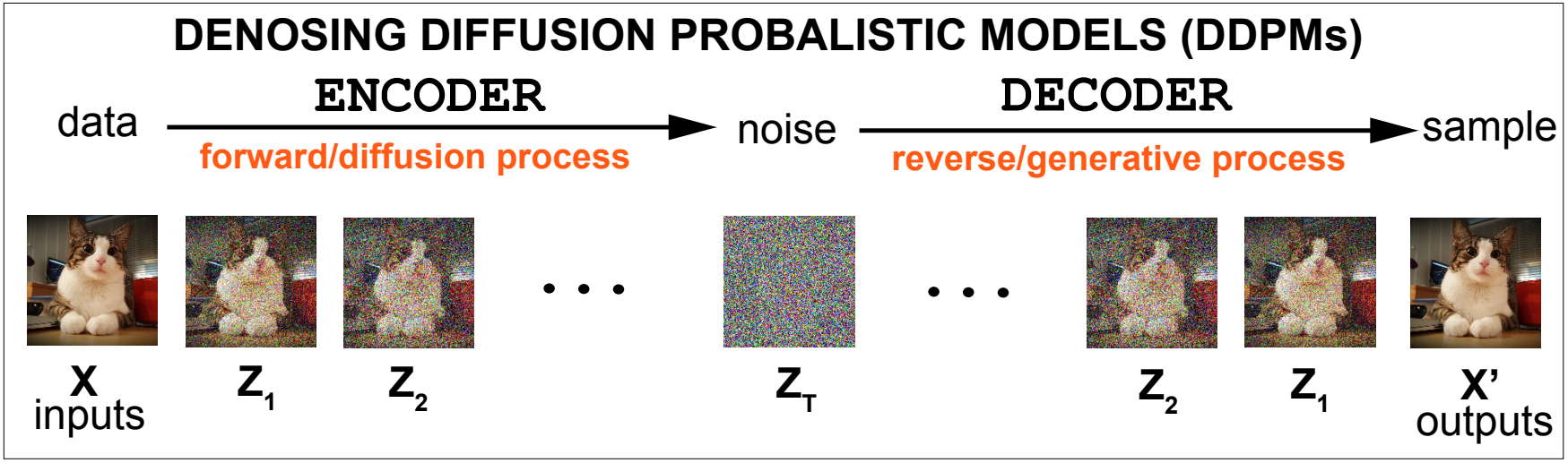
$P(Z_T|X) = N(0, I)$ $P(Z_T) = N(0, I)$

Encoder NOT a neural network


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
inputs
Z_1
Z_2
Z_T
Z_2
Z_1
X'
outputs

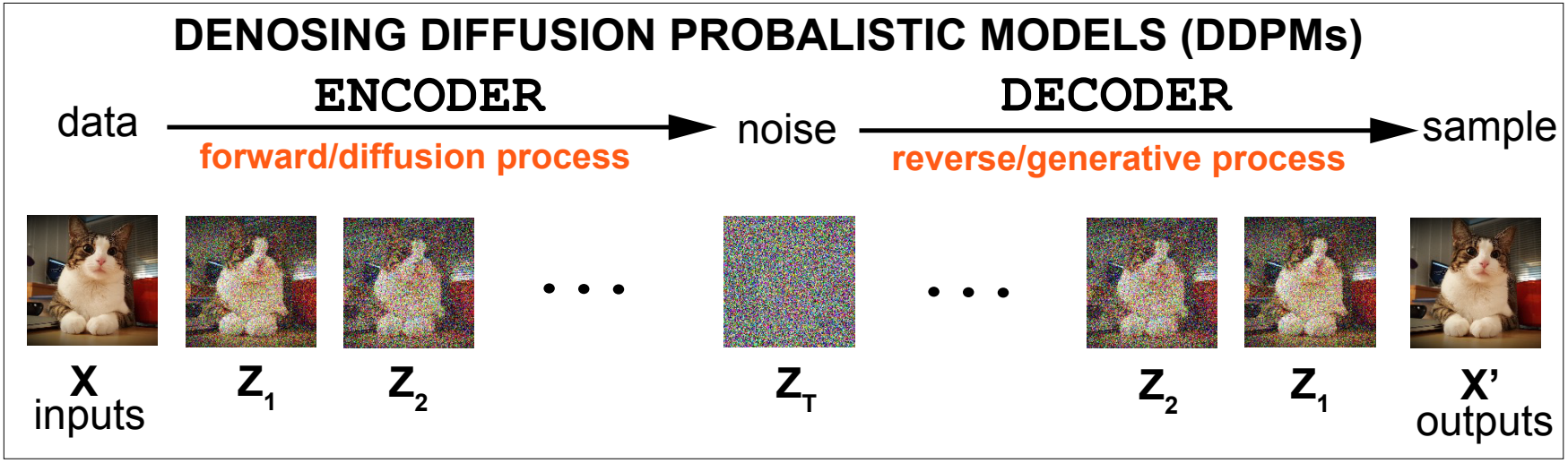


Decoder = reverse diffusion



Decoder = reverse diffusion

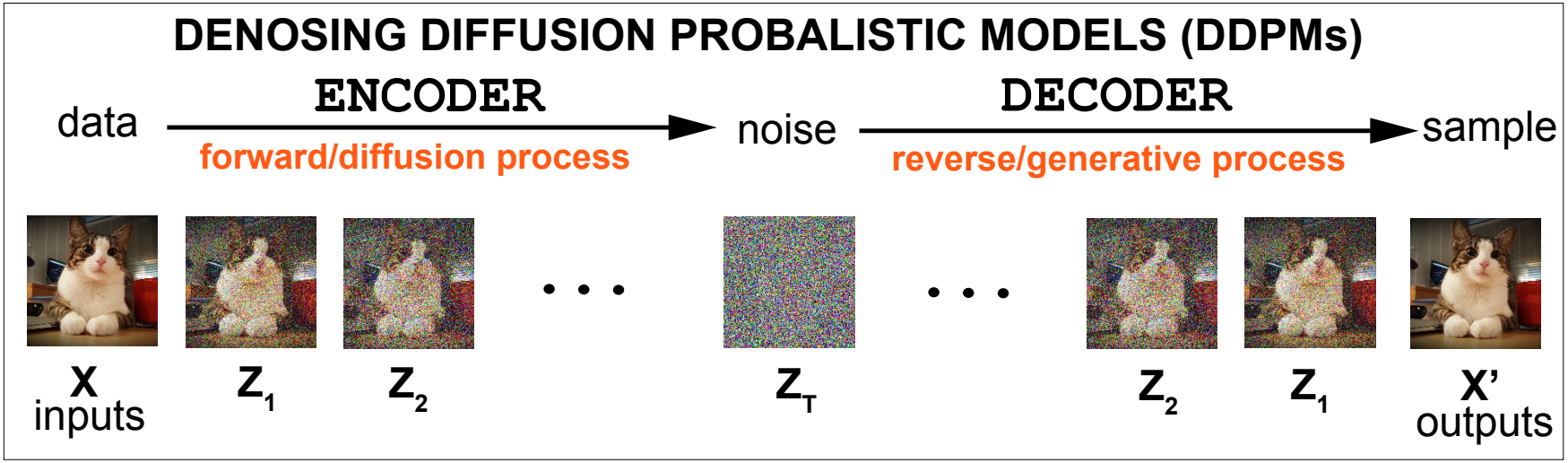
Start with Z_T just noise



Decoder = reverse diffusion

Start with Z_T just noise

$$Z_{T-1} = \mathsf{NN}_{\theta_T}(Z_T)$$

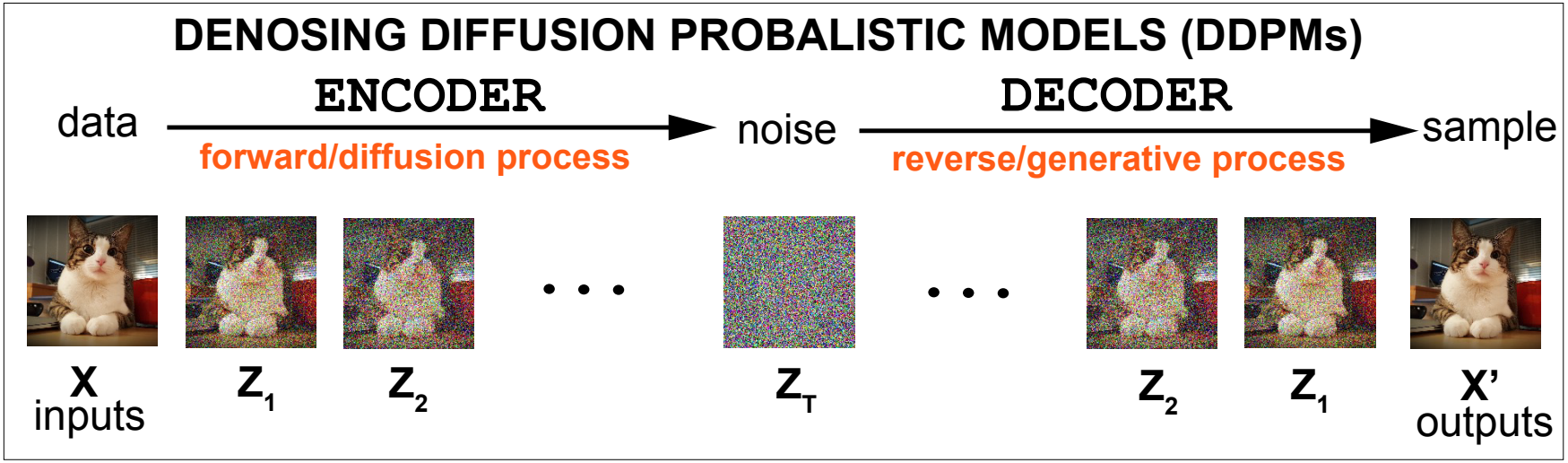


Decoder = reverse diffusion

Start with Z_T just noise

$$Z_{T-1} = \mathsf{NN}_{\theta_T}(Z_T)$$

$$Z_{T-2} = \mathsf{NN}_{\theta_{T-1}}(Z_{T-1})$$



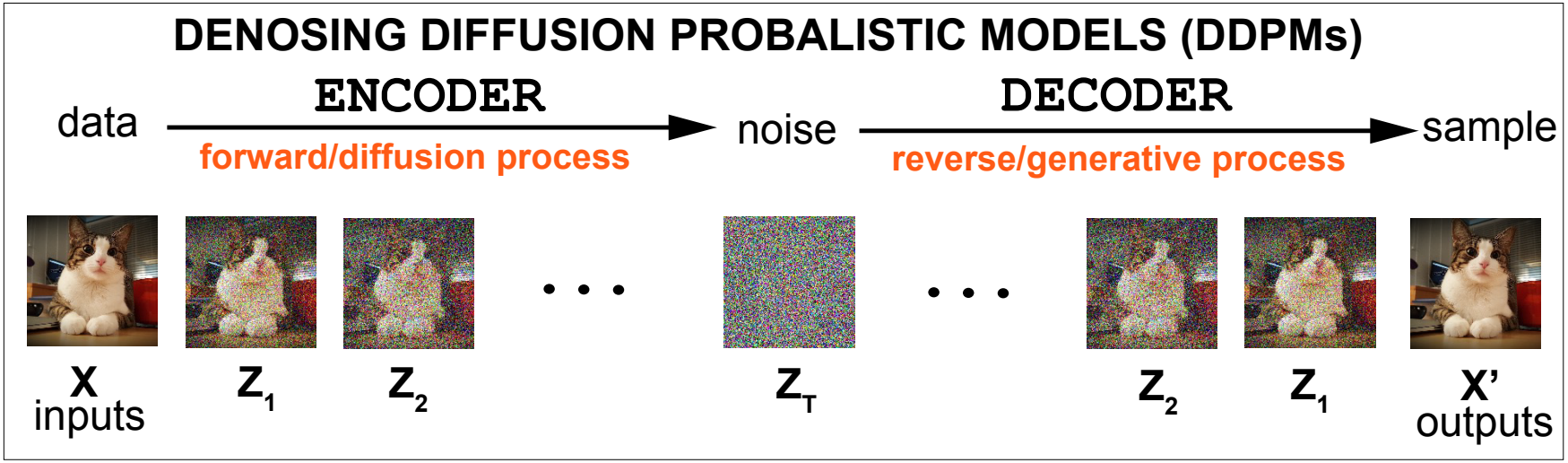
Decoder = reverse diffusion

Start with Z_T just noise

$$Z_{T-1} = \mathsf{NN}_{\theta_T}(Z_T)$$

$$Z_{T-2} = \mathsf{NN}_{\theta_{T-1}}(Z_{T-1})$$

...



Decoder = reverse diffusion

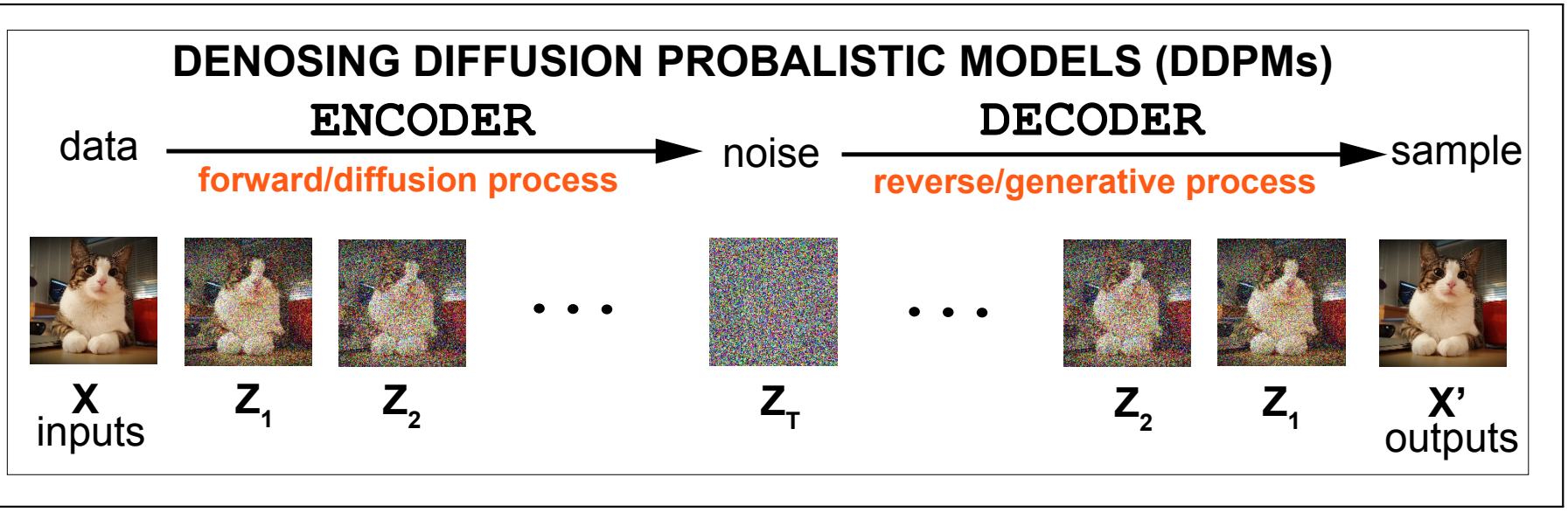
Start with Z_T just noise

$$Z_{T-1} = \mathsf{NN}_{\theta_T}(Z_T)$$

$$Z_{T-2} = \mathsf{NN}_{\theta_{T-1}}(Z_{T-1})$$

...

new data value: $X' = \mathsf{NN}_{\theta_1}(Z_1)$



Decoder = reverse diffusion

Start with Z_T just noise

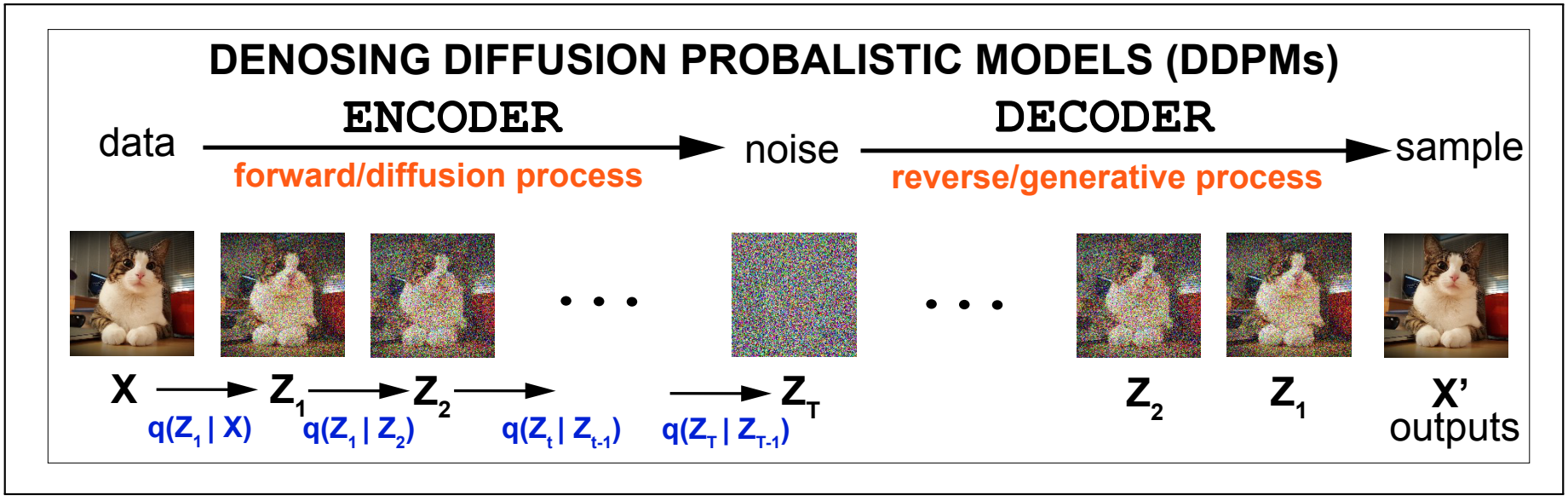
$$Z_{T-1} = \mathsf{NN}_{\theta_T}(Z_T)$$

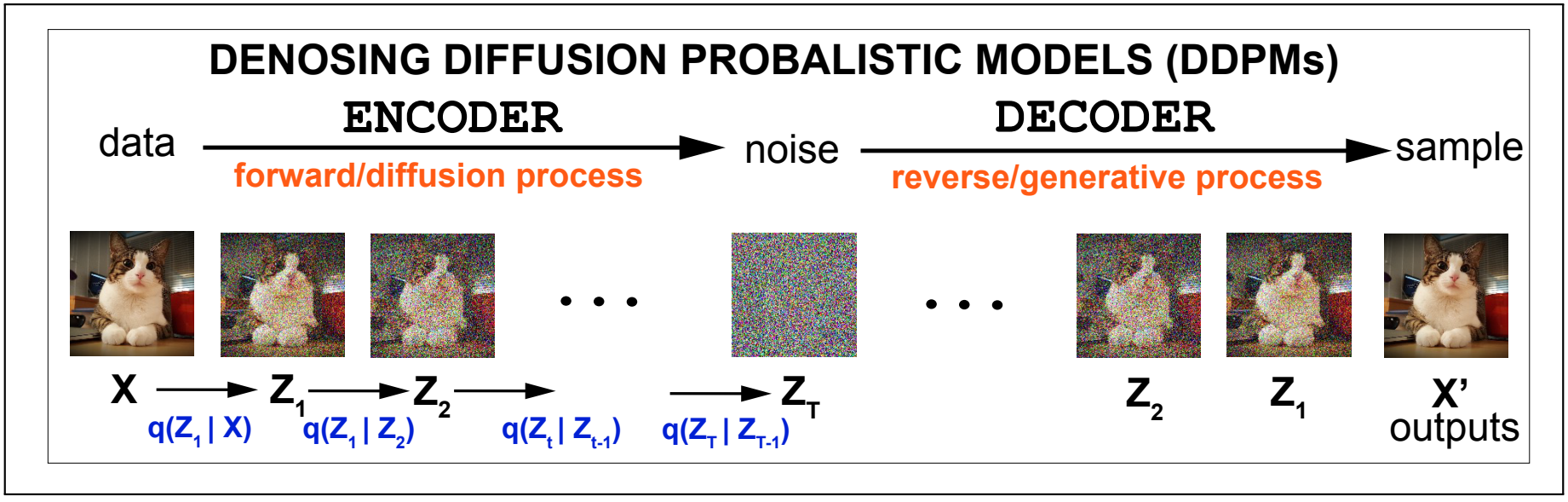
$$Z_{T-2} = \mathsf{NN}_{\theta_{T-1}}(Z_{T-1})$$

...

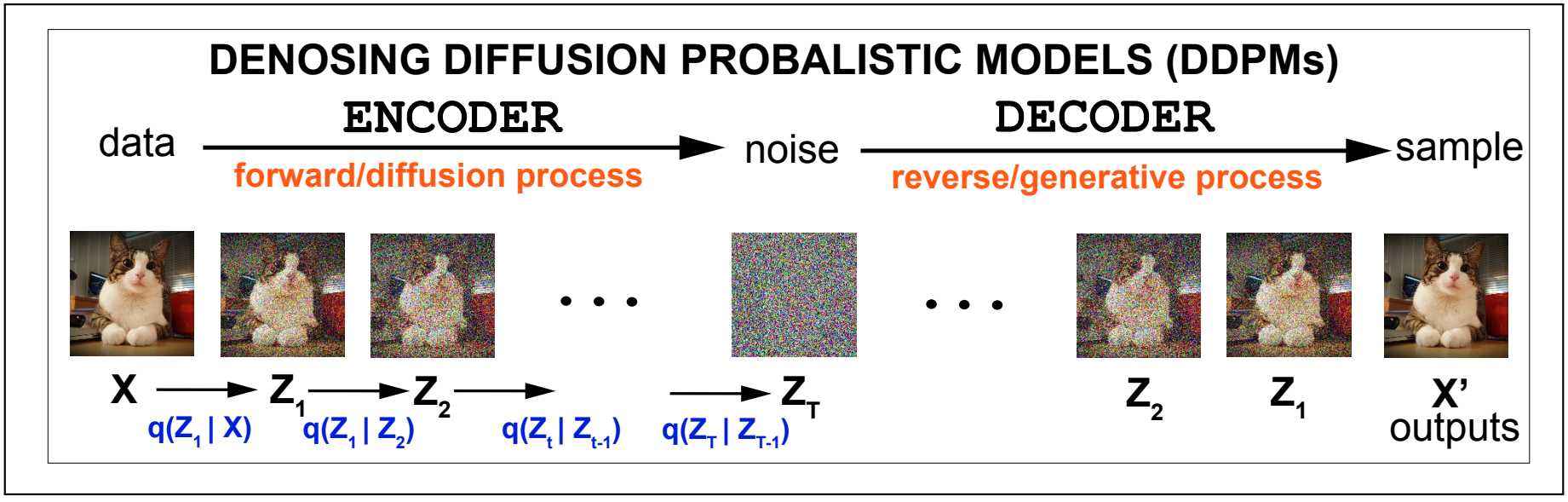
new data value: $X' = \mathsf{NN}_{\theta_1}(Z_1)$

One Neural Network per time step


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
Z_1
Z_2
Z_T
Z_2
Z_1
X'
outputs
$q(Z_1 | X)$
$q(Z_1 | Z_2)$
$q(Z_t | Z_{t-1})$
$q(Z_T | Z_{T-1})$

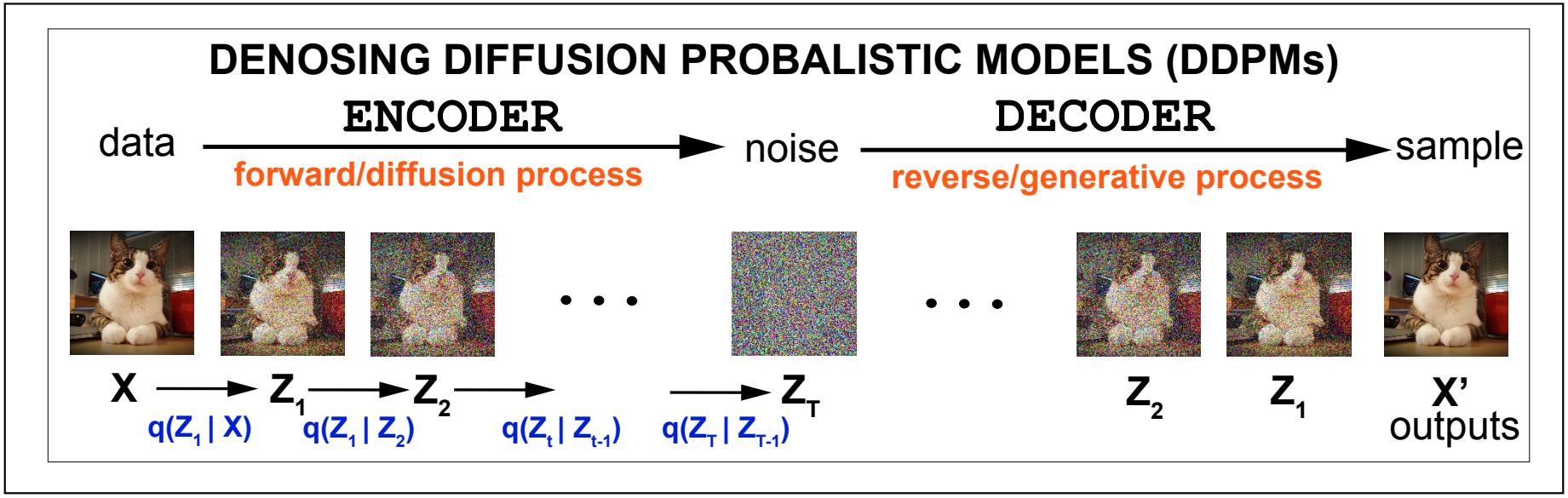


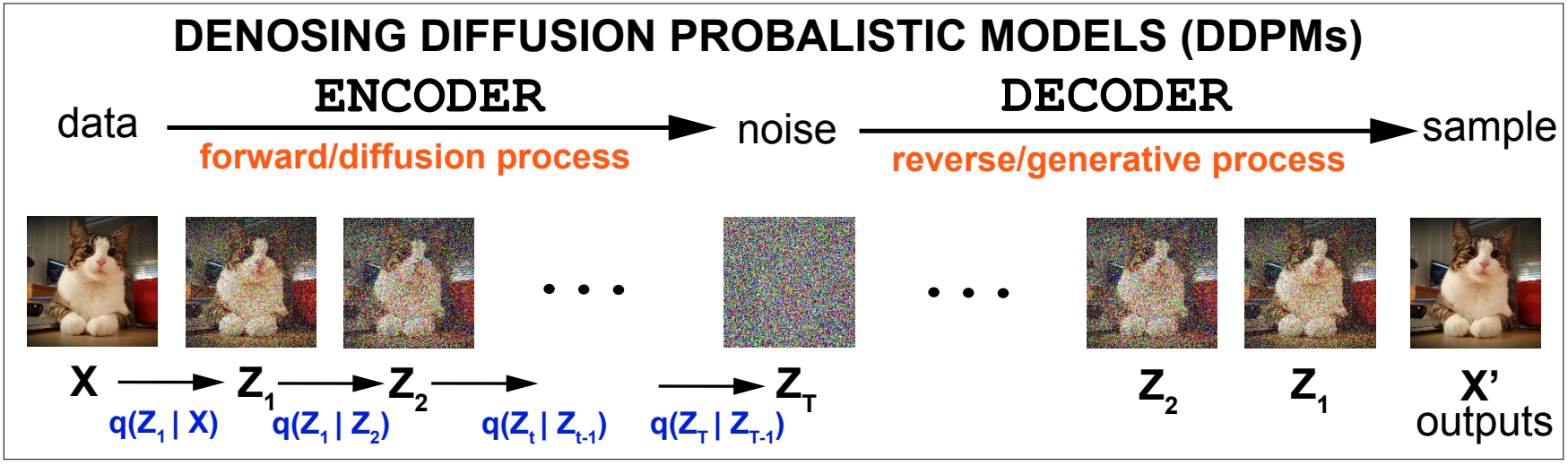
Diffusion is Probabilistic



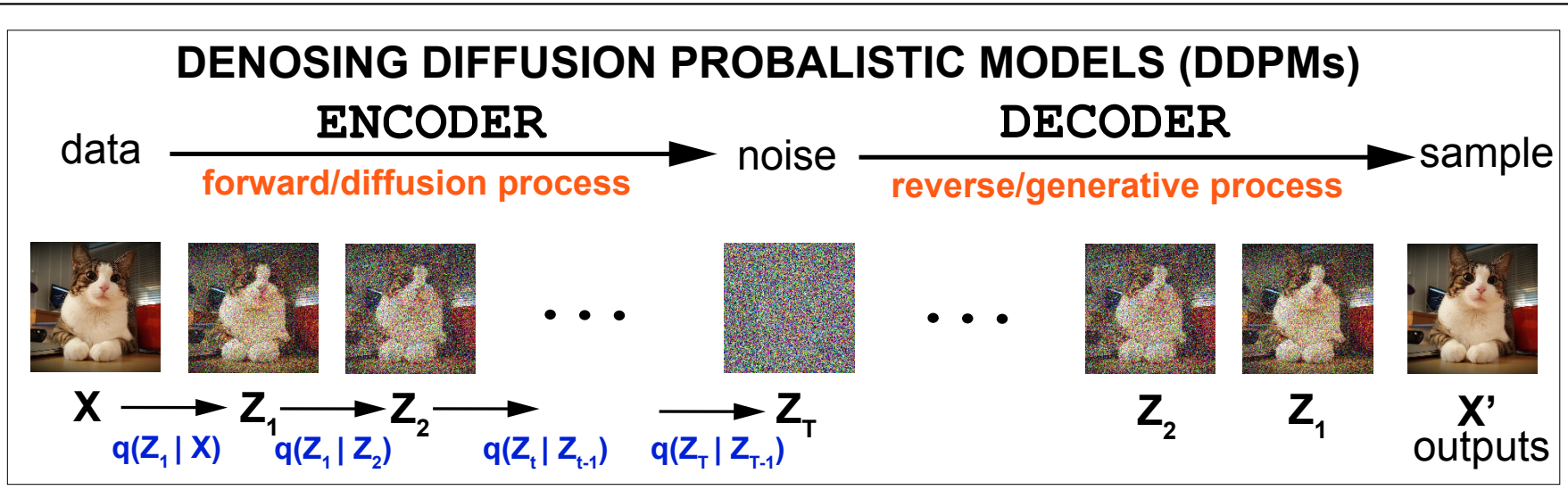
Diffusion is Probabilistic

$$q(Z_1 \dots Z_T \mid X)$$


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
Z1
Z2
ZT
Z2
Z1
X'
outputs
q(Z1 | X)
q(Z1 | Z2)
q(Zt | Zt-1)
q(ZT | ZT-1)

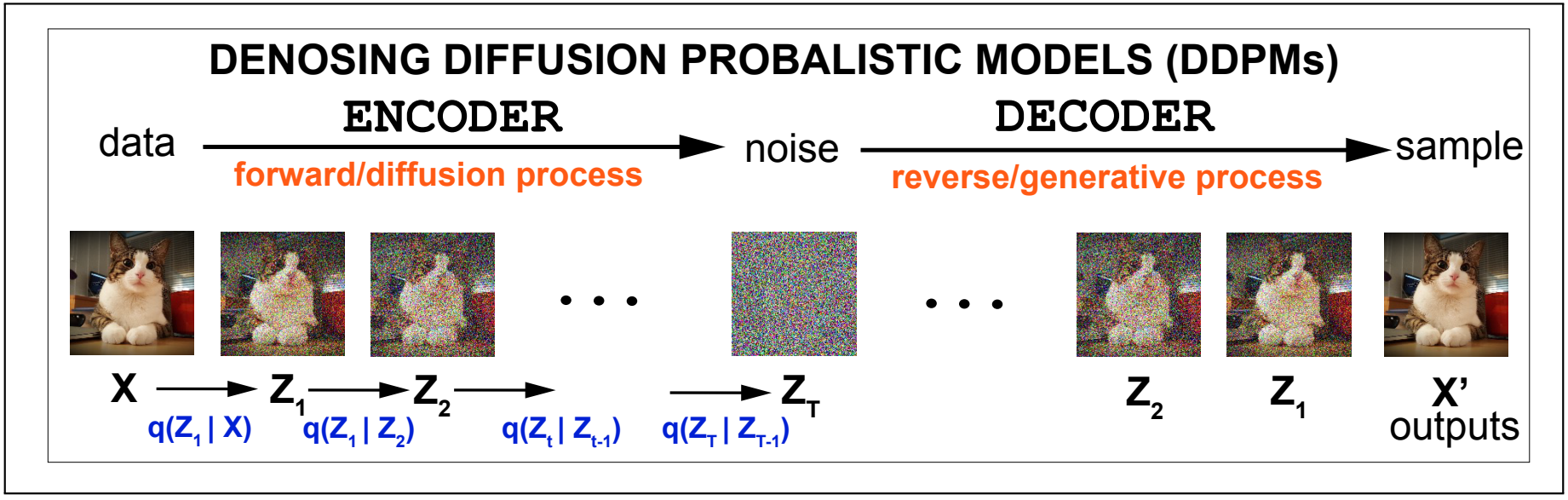


Diffusion is a Markov process



Diffusion is a Markov process

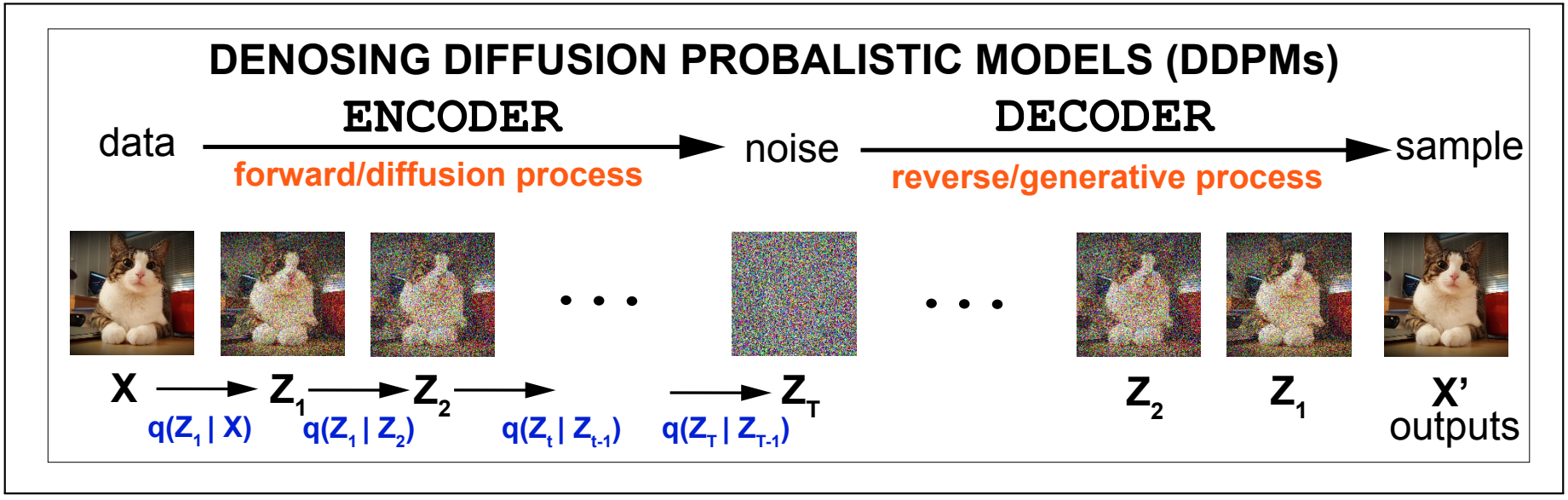
$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) q(Z_2 \mid Z_1) \dots q(Z_T \mid Z_{T-1})$$



Diffusion is a Markov process

$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) q(Z_2 \mid Z_1) \dots q(Z_T \mid Z_{T-1})$$

$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) \prod_{t=2}^{T} q(Z_t \mid Z_{t-1})$$

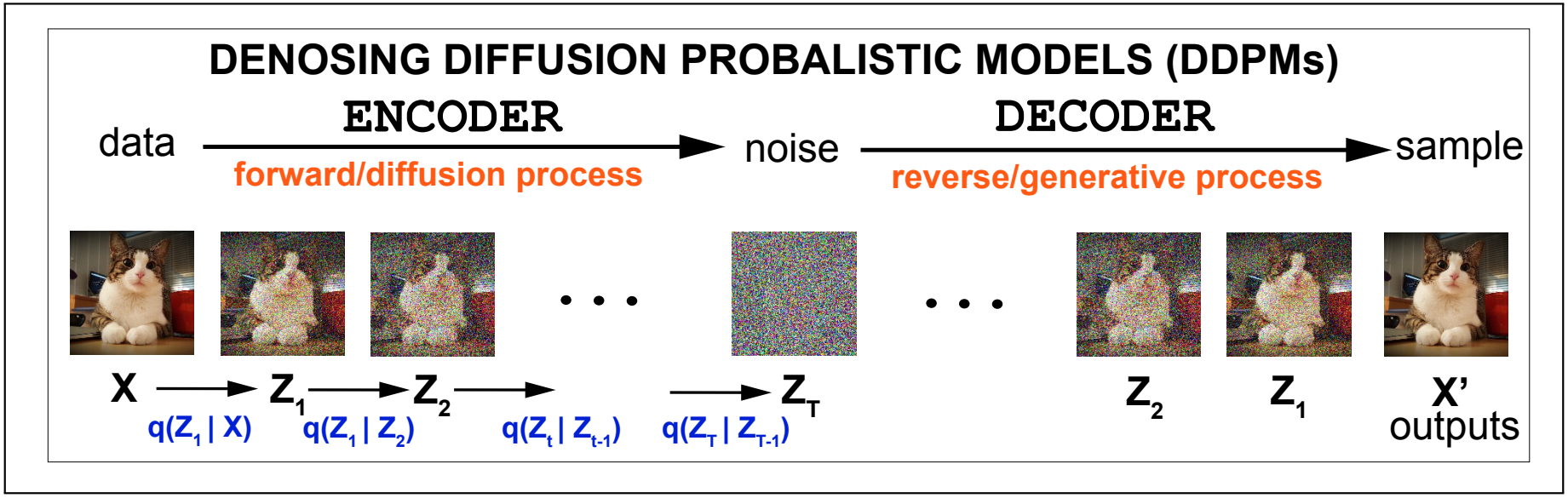


Diffusion is a Markov process

$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) q(Z_2 \mid Z_1) \dots q(Z_T \mid Z_{T-1})$$

$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) \prod_{t=2}^{T} q(Z_t \mid Z_{t-1})$$

$$q(Z_t \mid Z_{t-1}) \text{ ??}$$



Diffusion is a Markov process

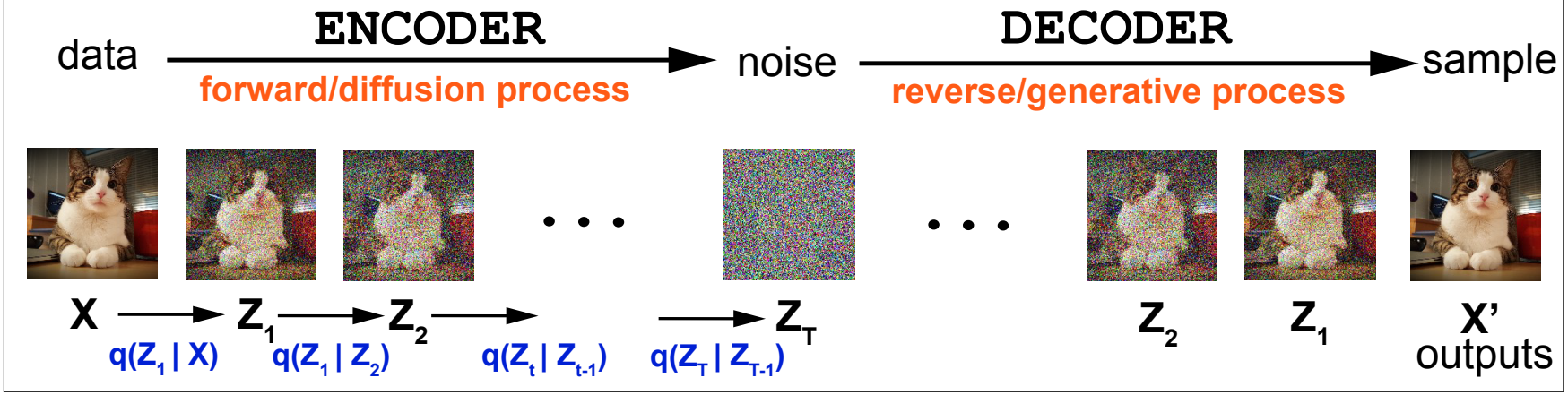
$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) q(Z_2 \mid Z_1) \dots q(Z_T \mid Z_{T-1})$$

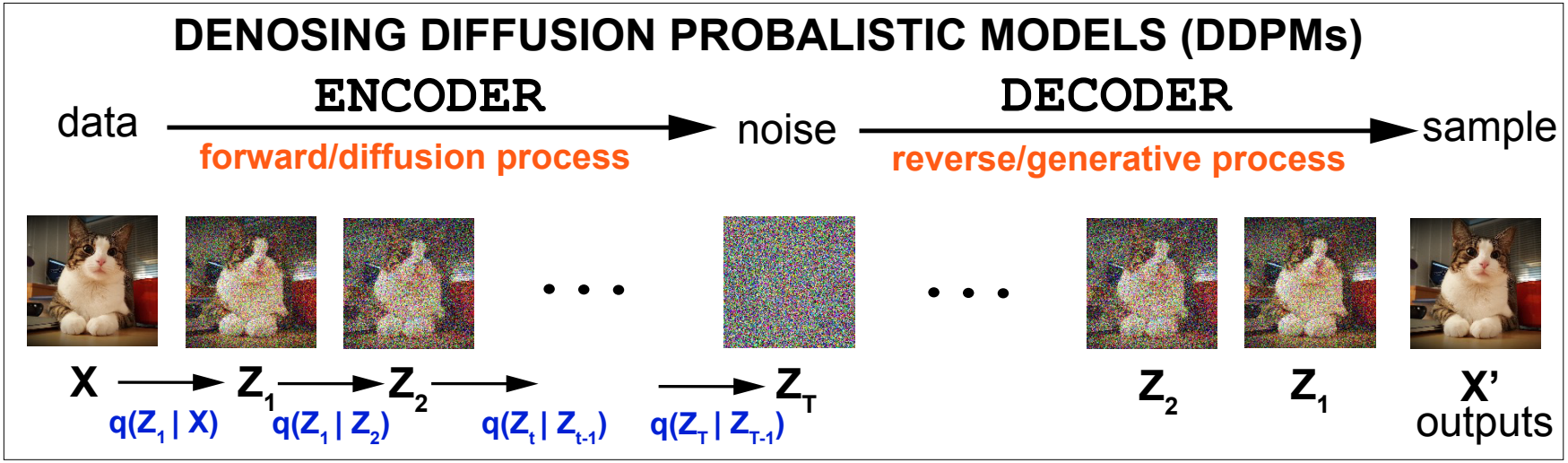
$$q(Z_1 \dots Z_T \mid X) = q(Z_1 \mid X) \prod_{t=2}^{T} q(Z_t \mid Z_{t-1})$$

$$q(Z_t \mid Z_{t-1}) \text{ ??}$$

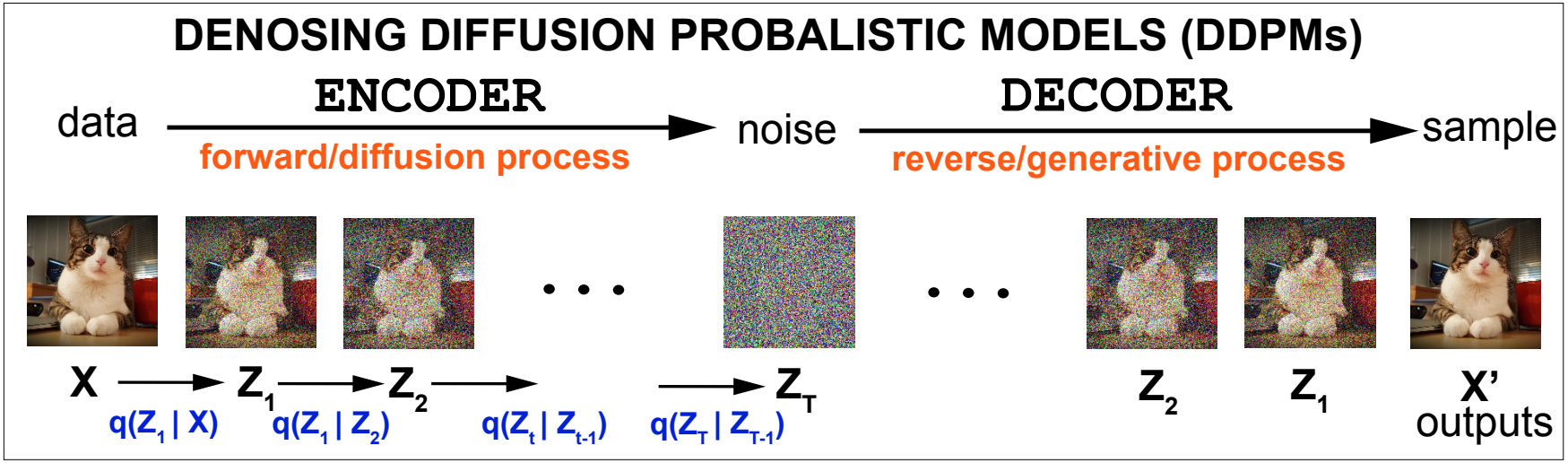
The Encoder "schedule"

DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)



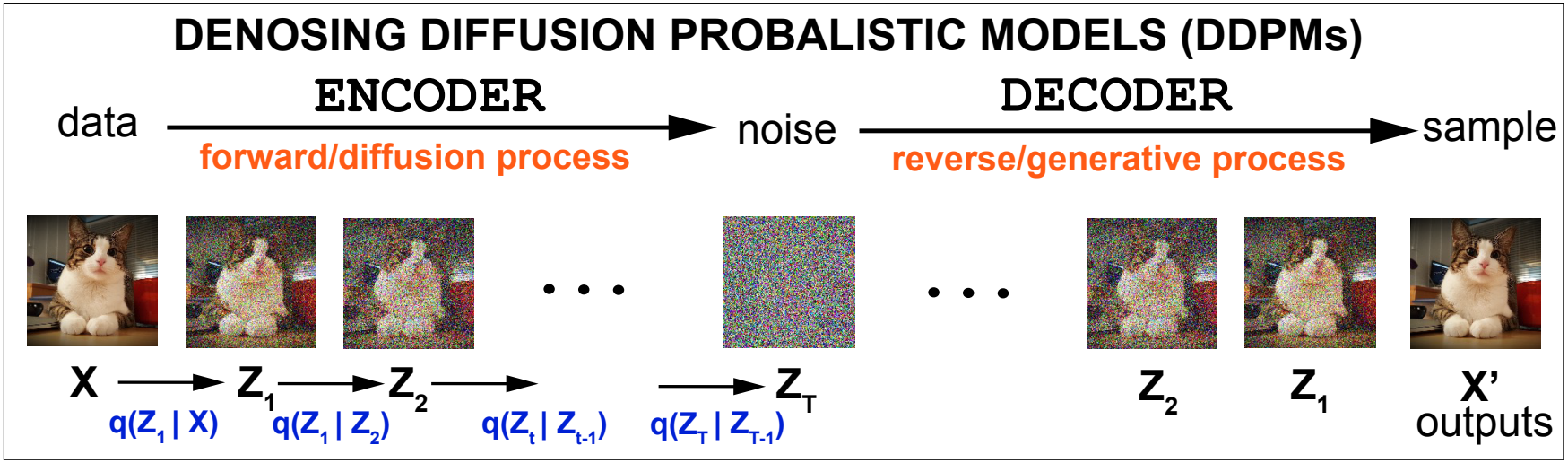


Diffusion adds noise... slowly



Diffusion adds noise... slowly

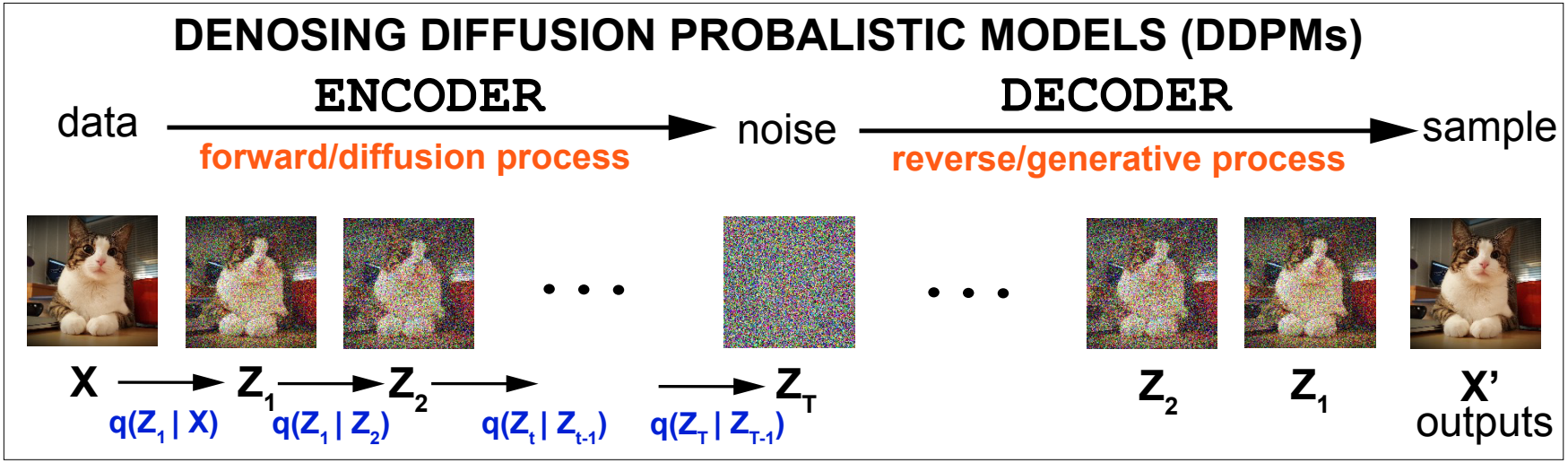
$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$



Diffusion adds noise... slowly

$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$

$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

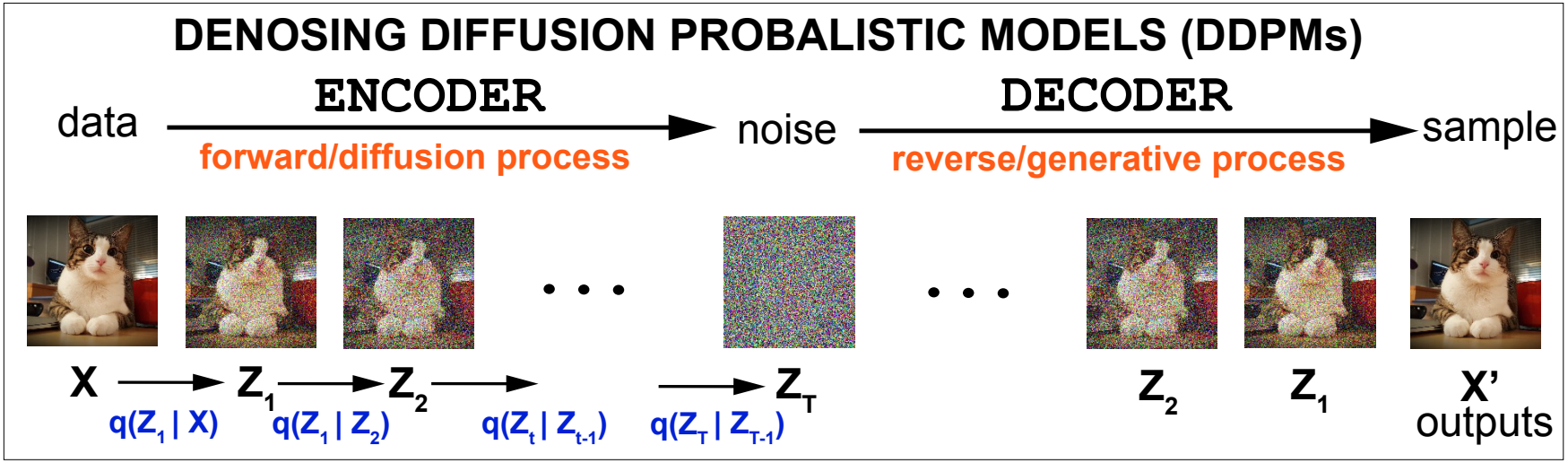


Diffusion adds noise... slowly

$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$

$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

$$Z_t = \sqrt{1-\beta_t}\, Z_{t-1} + \sqrt{\beta_t}\, \epsilon_t$$



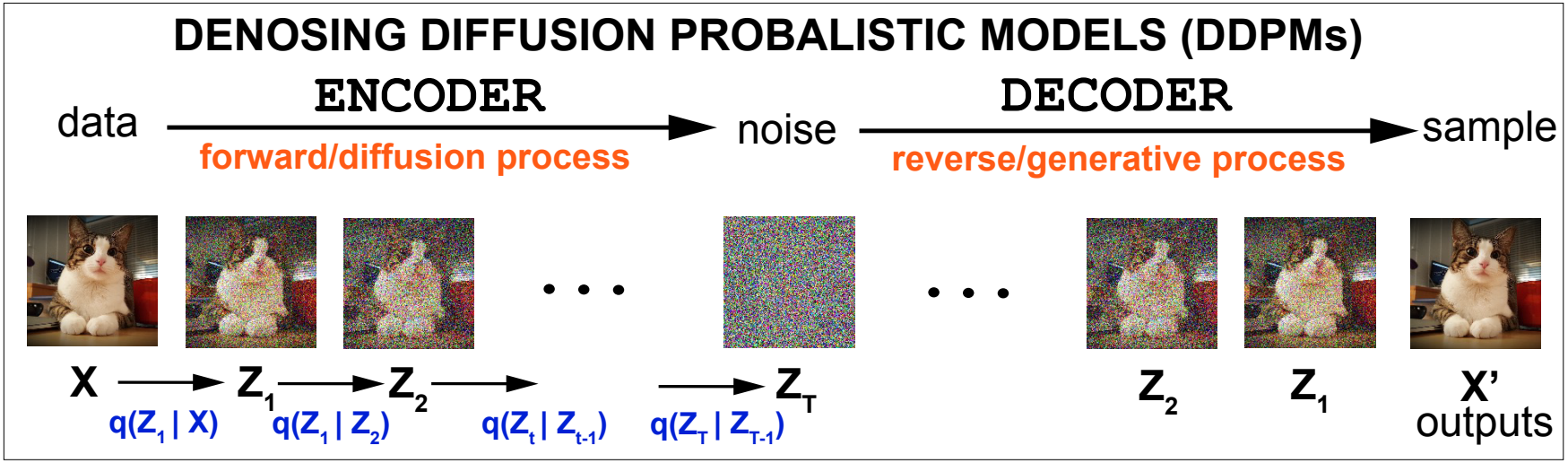
Diffusion adds noise... slowly

$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$

$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

$$Z_t = \sqrt{1-\beta_t}\, Z_{t-1} + \sqrt{\beta_t}\, \epsilon_t$$

$0 \leq \beta_1, \beta_2, \ldots, \beta_T \leq 1$ meta-parameters



Diffusion adds noise... slowly

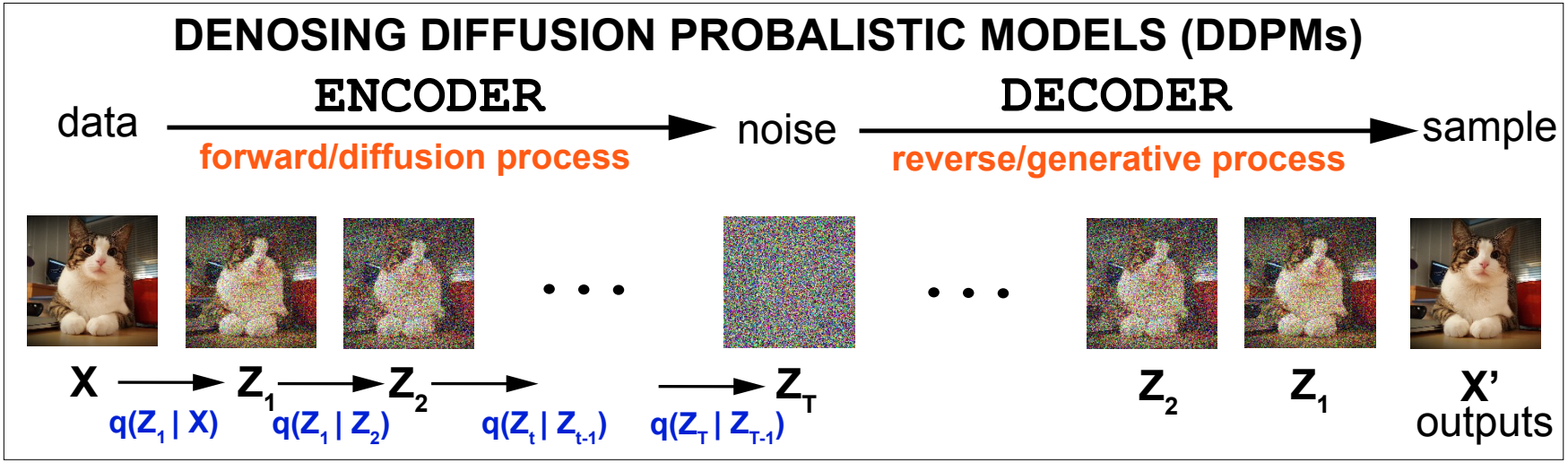
$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$

$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

$$Z_t = \sqrt{1-\beta_t}\, Z_{t-1} + \sqrt{\beta_t}\, \epsilon_t$$

$0 \leq \beta_1, \beta_2, \ldots, \beta_T \leq 1$ meta-parameters

$$\epsilon_t \sim N(0, I)$$



Diffusion adds noise... slowly

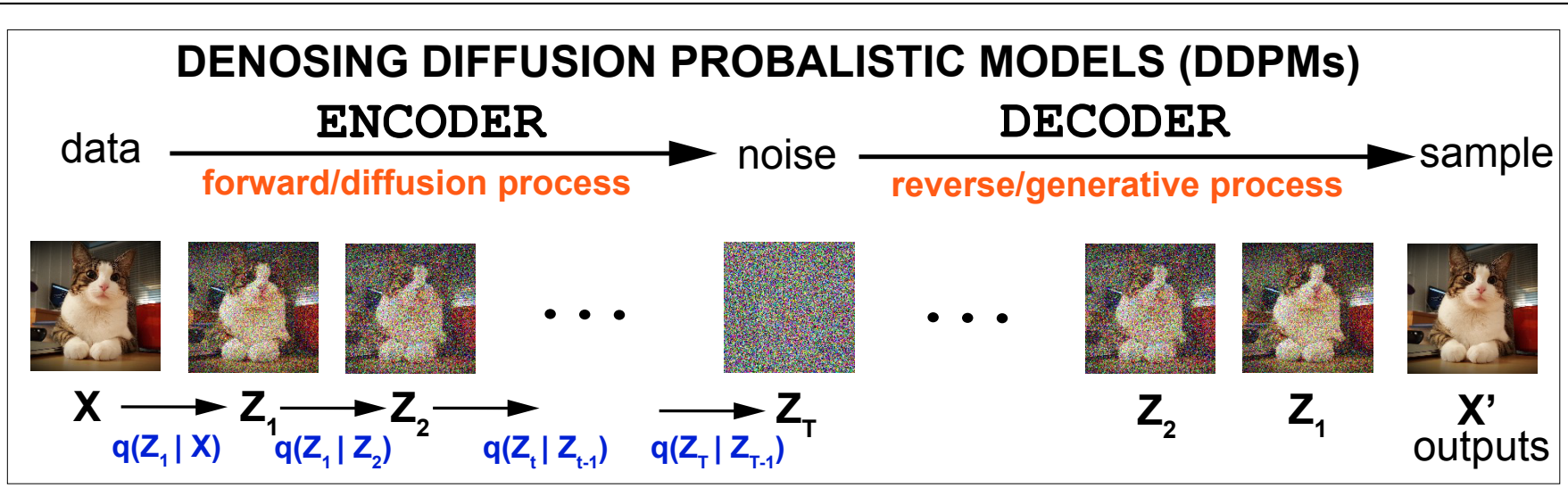
$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$

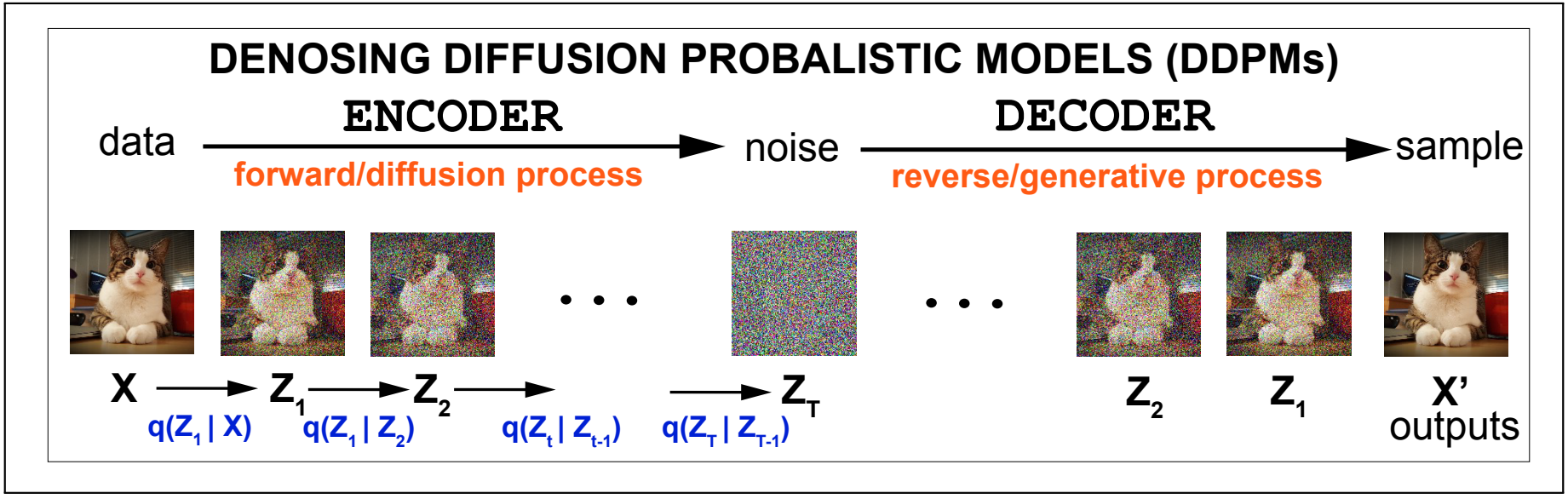
$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

$$Z_t = \sqrt{1-\beta_t}\, Z_{t-1} + \sqrt{\beta_t}\, \epsilon_t$$

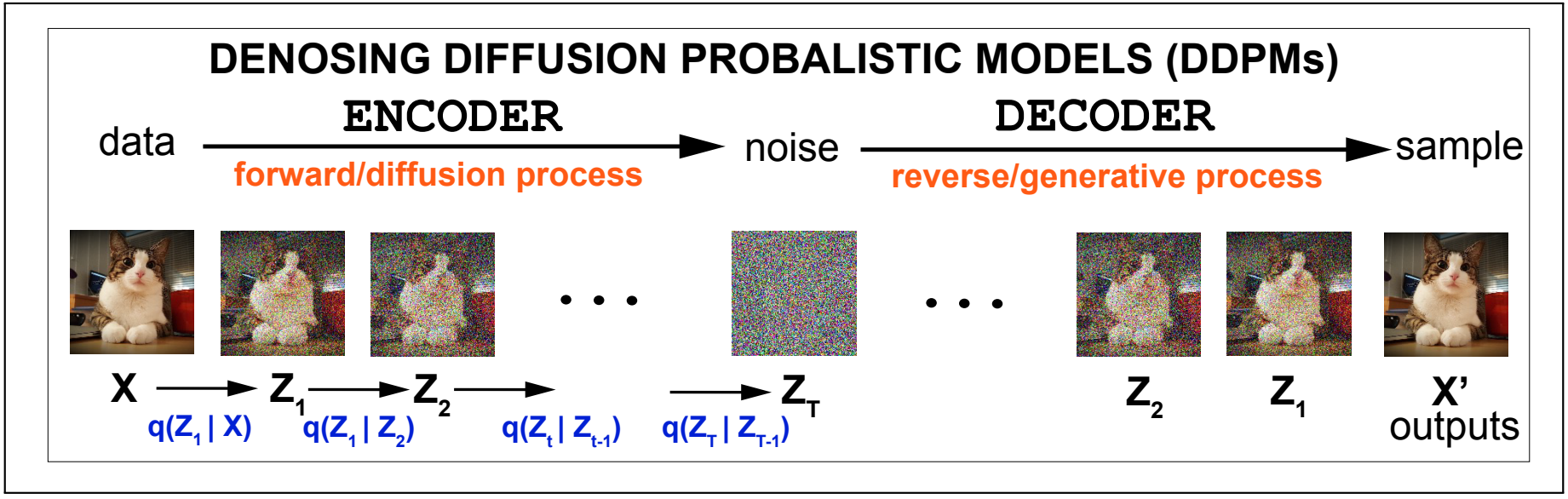
$0 \leq \beta_1, \beta_2, \ldots, \beta_T \leq 1$ meta-parameters

$$\epsilon_t \sim N(0, I)$$


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
Z_1
Z_2
Z_T
Z_2
Z_1
X'
outputs
$q(Z_1 | X)$
$q(Z_1 | Z_2)$
$q(Z_t | Z_{t-1})$
$q(Z_T | Z_{T-1})$

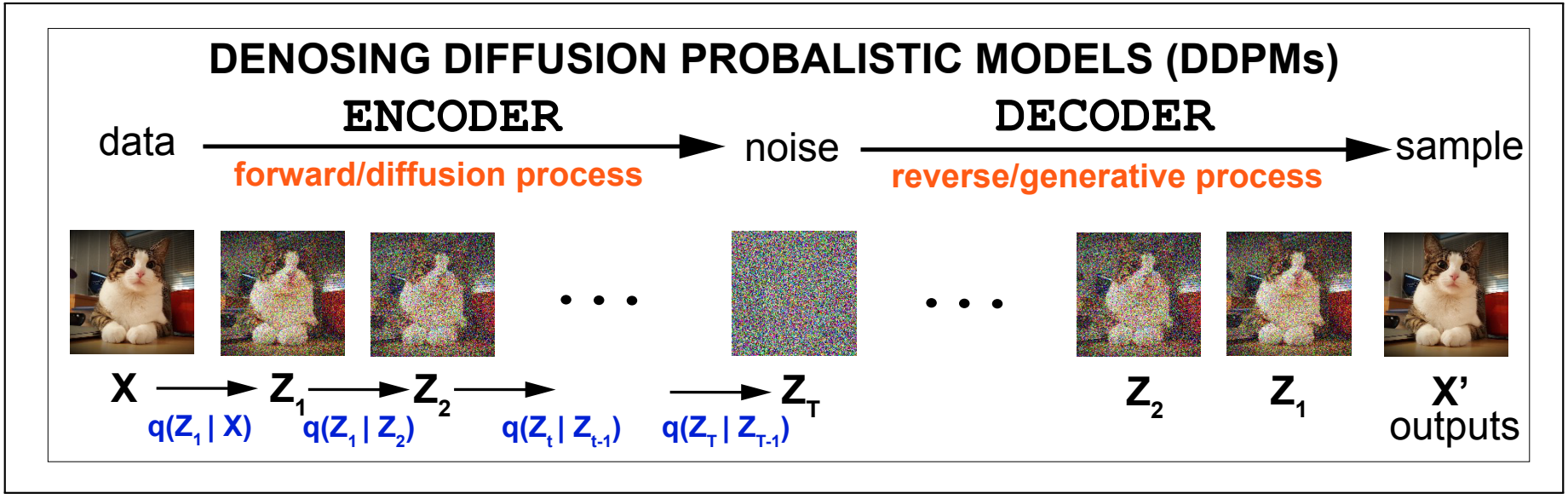


Diffusion adds noise... slowly



Diffusion adds noise... slowly

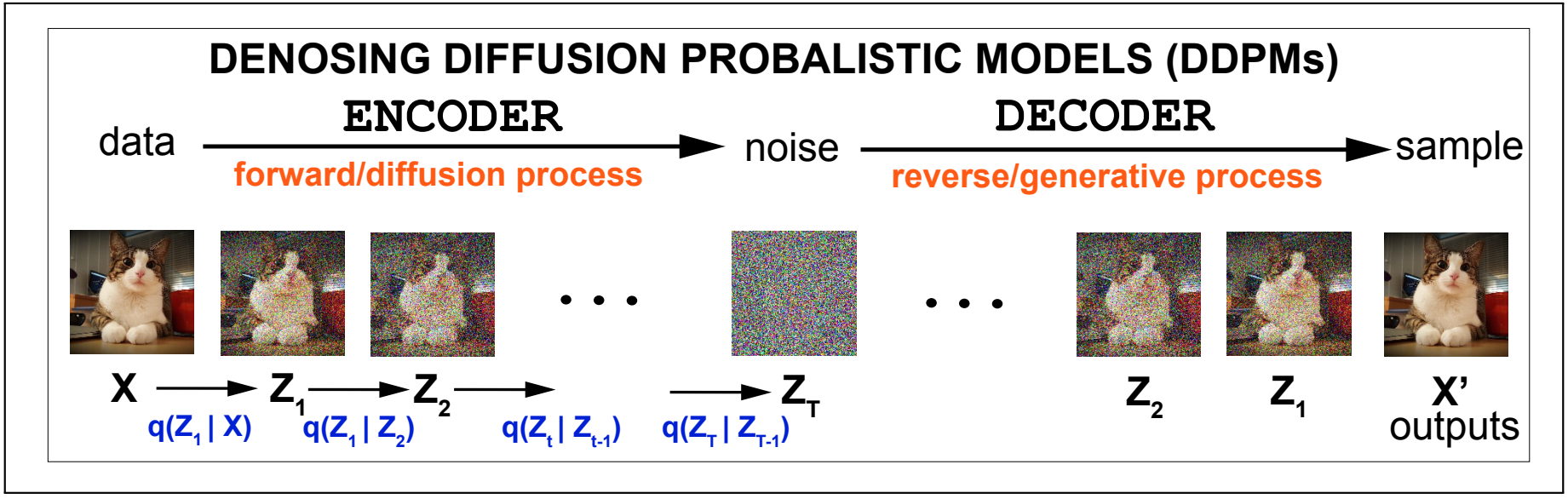
$$q(Z_1 \mid X) = N(\sqrt{1-\beta_1}\, X, \beta_1\, I)$$



Diffusion adds noise... slowly

$$q(Z_1 \mid X) = N(\sqrt{1-\beta_1}\, X, \beta_1\, I)$$

$$q(Z_2 \mid Z_1) = N(\sqrt{1-\beta_2}\, Z_1, \beta_2\, I)$$

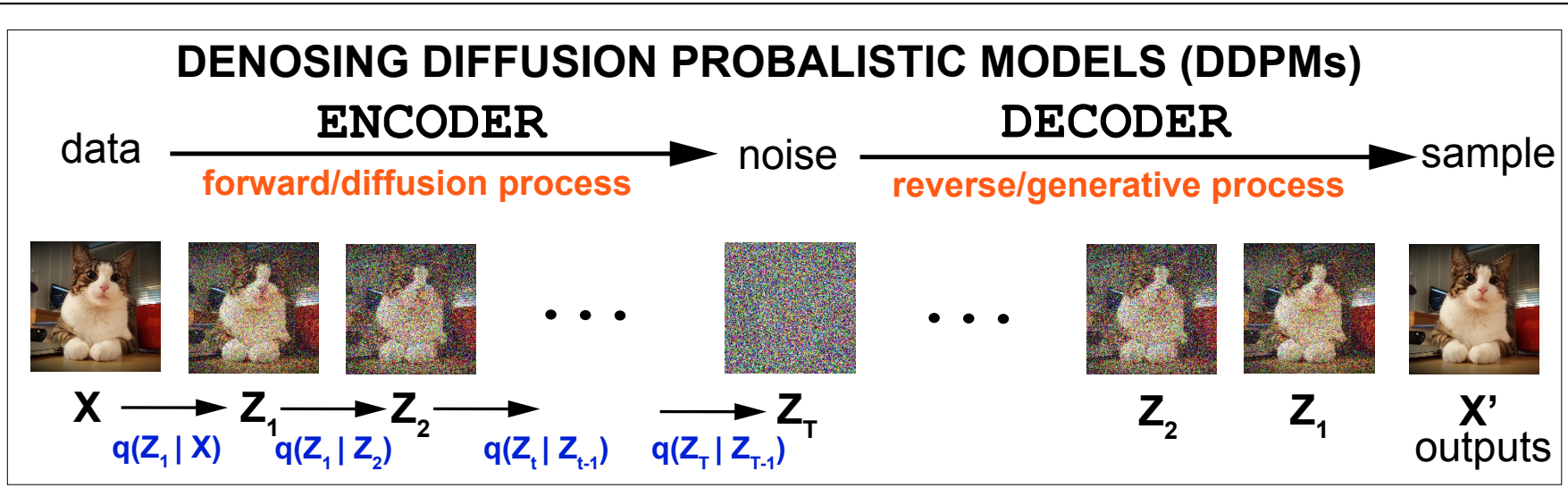


Diffusion adds noise... slowly

$$q(Z_1 \mid X) = N(\sqrt{1-\beta_1}\, X, \beta_1\, I)$$

$$q(Z_2 \mid Z_1) = N(\sqrt{1-\beta_2}\, Z_1, \beta_2\, I)$$

$$q(Z_t \mid Z_{t-1}) = N(\sqrt{1-\beta_t}\, Z_{t-1}, \beta_t\, I)$$



Diffusion adds noise... slowly

$$q(Z_1 \mid X) = N(\sqrt{1-\beta_1}\, X, \beta_1\, I)$$

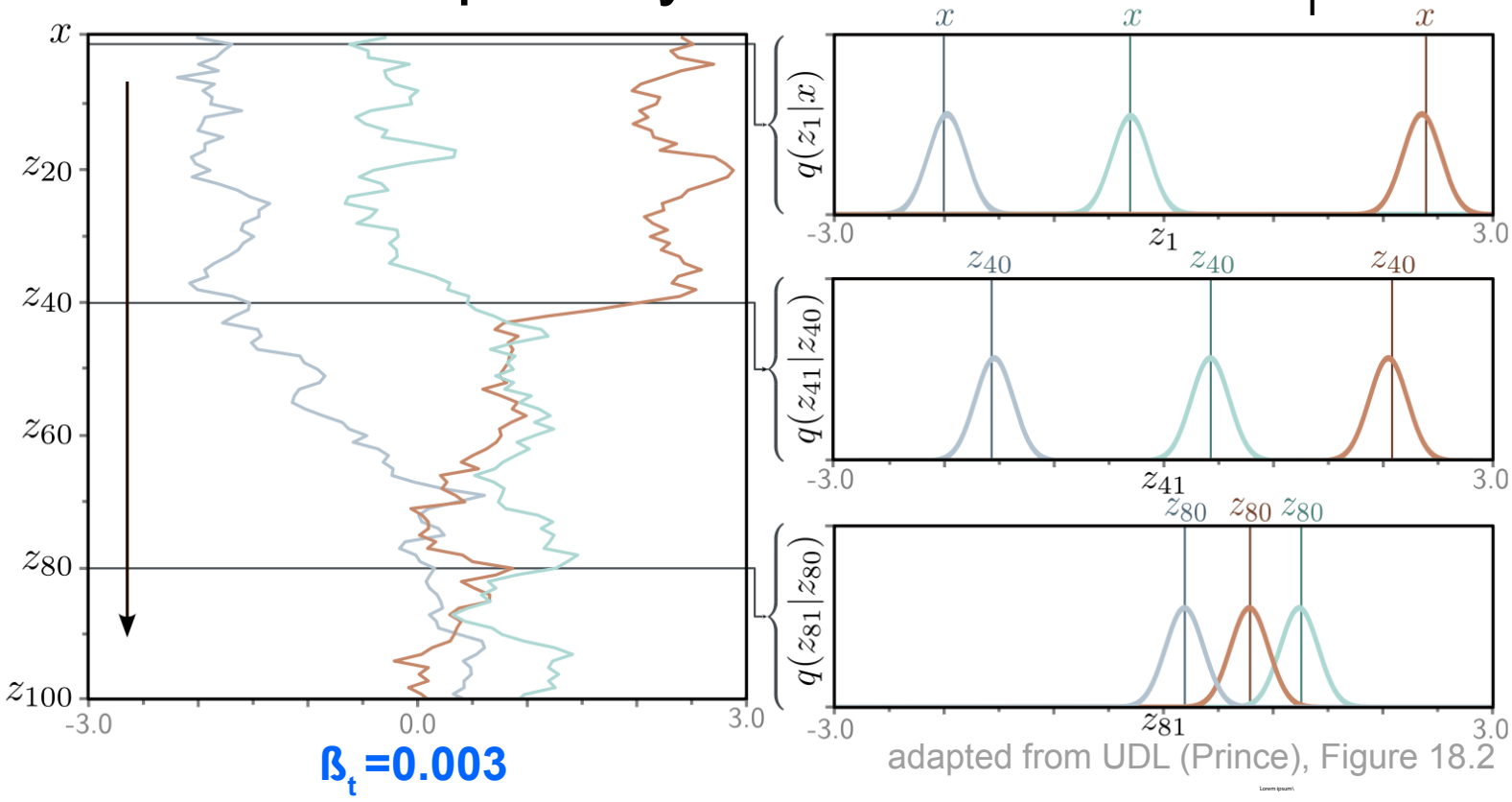
$$q(Z_2 \mid Z_1) = N(\sqrt{1-\beta_2}\, Z_1, \beta_2\, I)$$

$$q(Z_t \mid Z_{t-1}) = N(\sqrt{1-\beta_t}\, Z_{t-1}, \beta_t\, I)$$

the diffusion schedule

Diffusion Forward

3 example trayectories from x to z_T

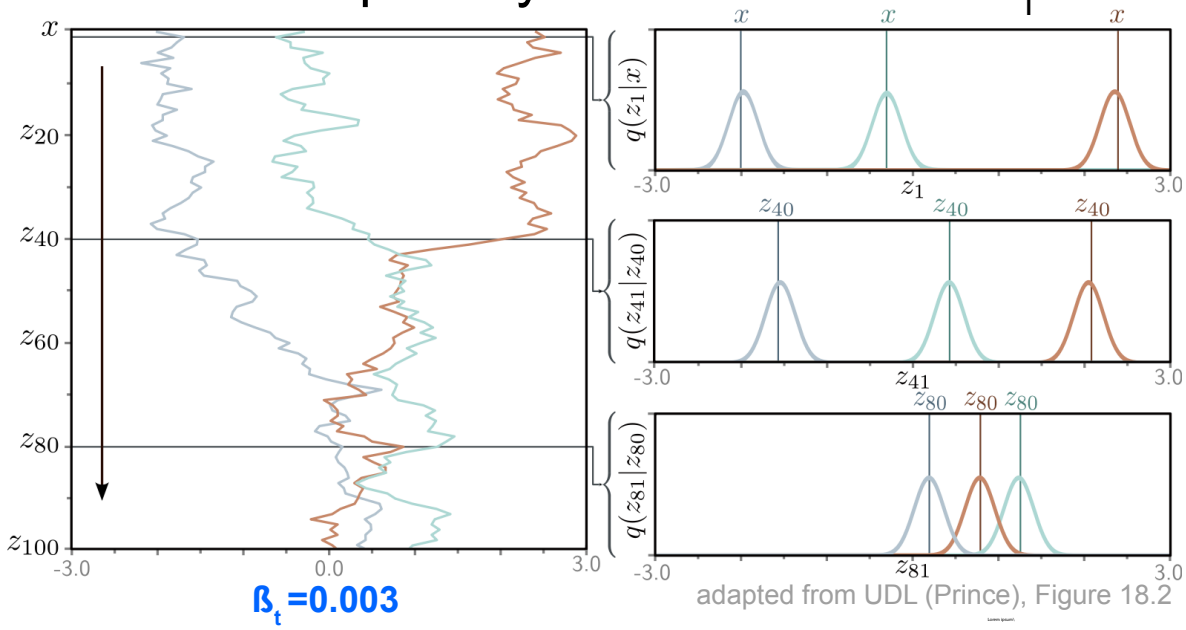


$ß_t$ =0.003

adapted from UDL (Prince), Figure 18.2

Diffusion Forward

3 example trayectories from x to z_T



$ß_t = 0.003$

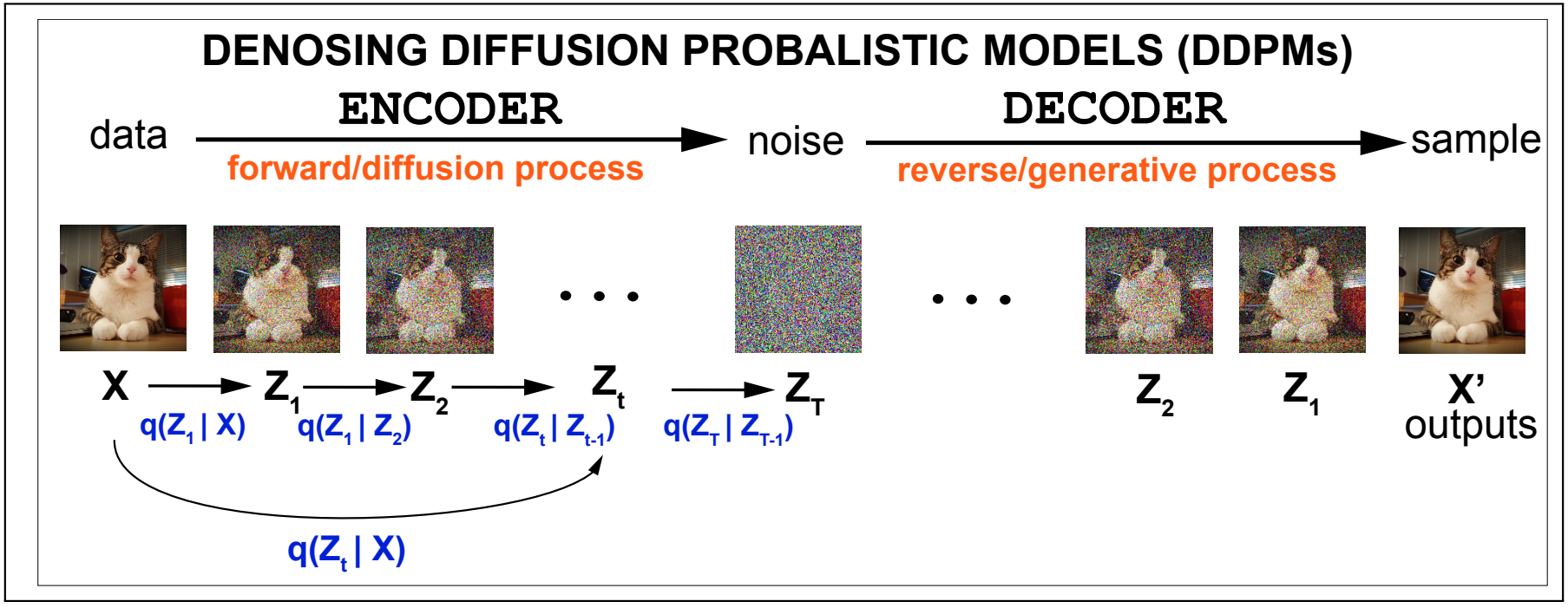
adapted from UDL (Prince), Figure 18.2

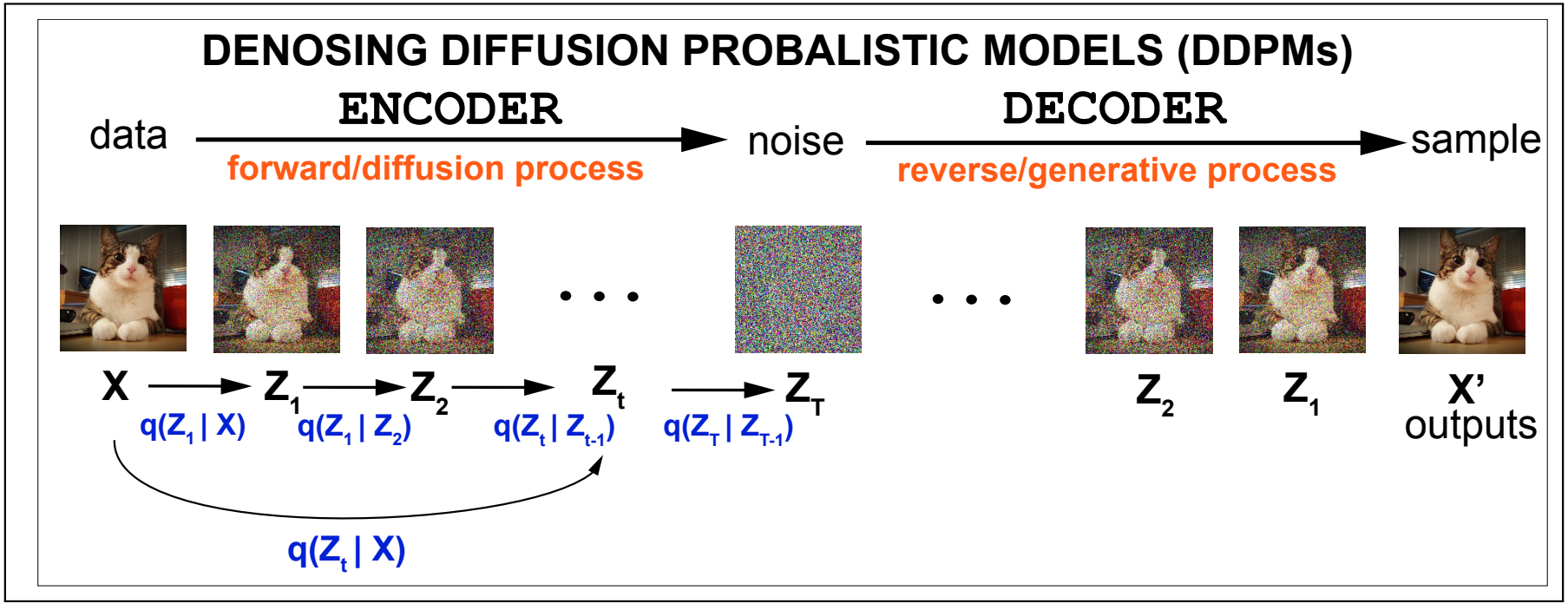
data → **ENCODER** → noise

forward/diffusion process

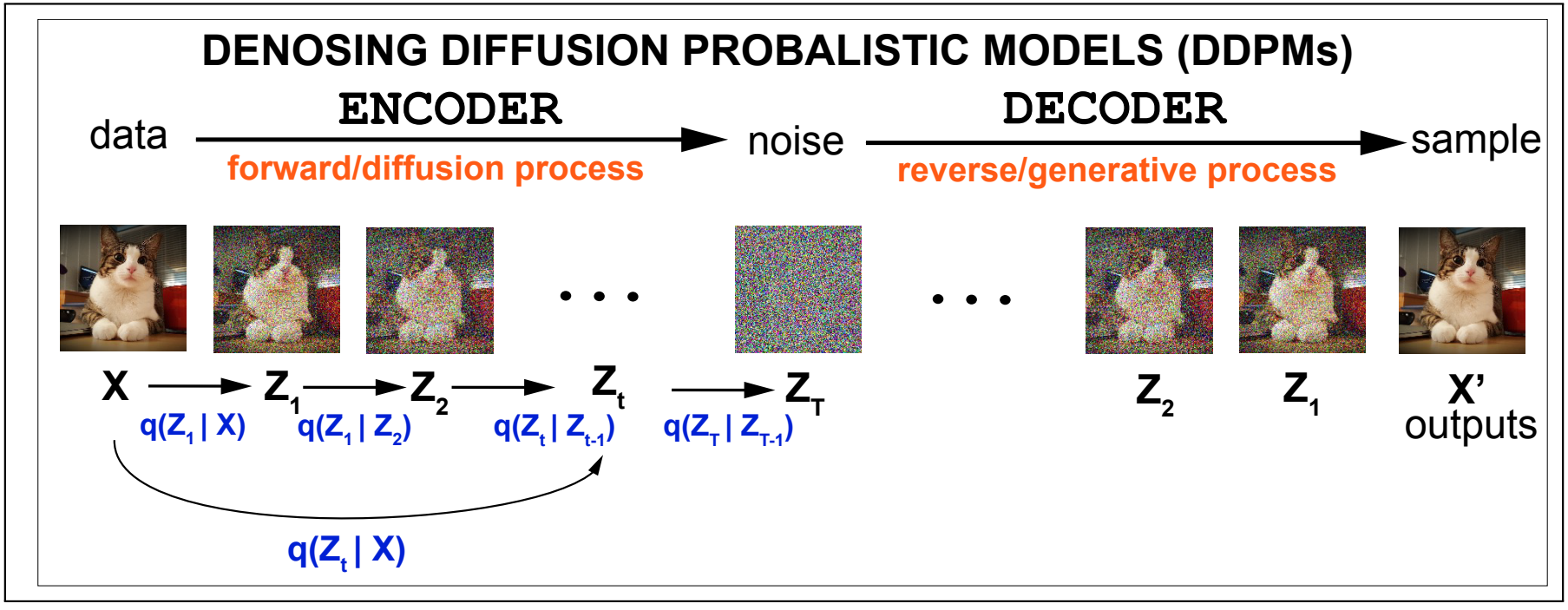
$$q(z_t|z_{t-1}) = N(z_t; \sqrt{1-ß_t}z_{t-1}, ß_t I)$$

$$X \rightarrow Z_1 \rightarrow Z_2 \rightarrow \cdots \rightarrow Z_{t-1} \rightarrow Z_t \rightarrow \cdots \rightarrow Z_T$$


DENOSING DIFFUSION PROBALISTIC MODELS (DDPMs)
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
Z_1
Z_2
Z_t
Z_T
Z_2
Z_1
X'
outputs
$q(Z_1 | X)$
$q(Z_1 | Z_2)$
$q(Z_t | Z_{t-1})$
$q(Z_T | Z_{T-1})$
$q(Z_t | X)$

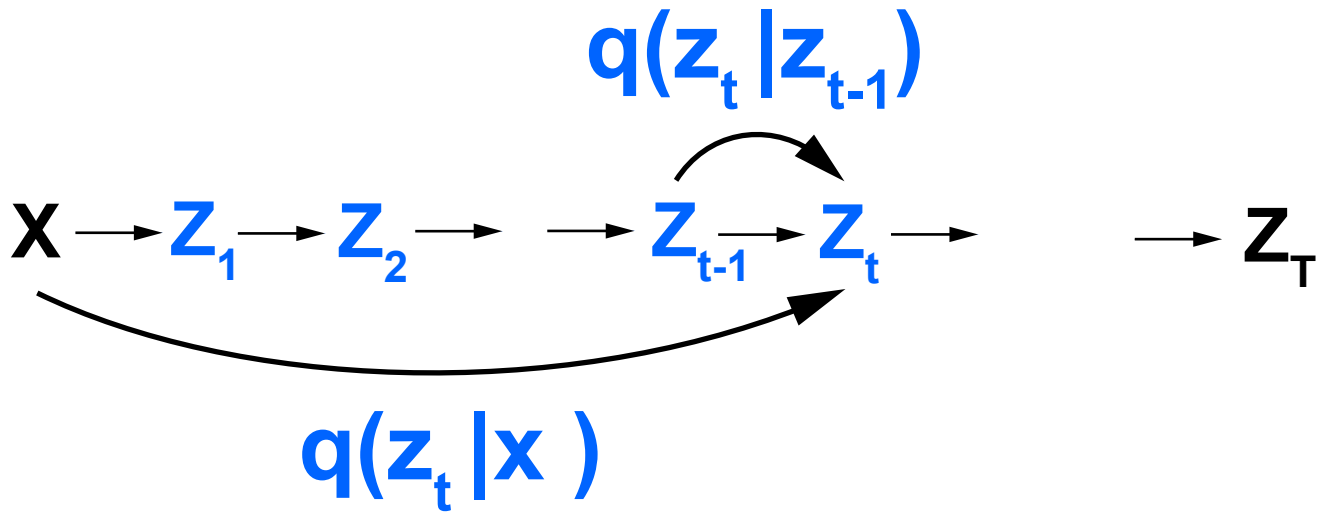


Diffusion Kernel

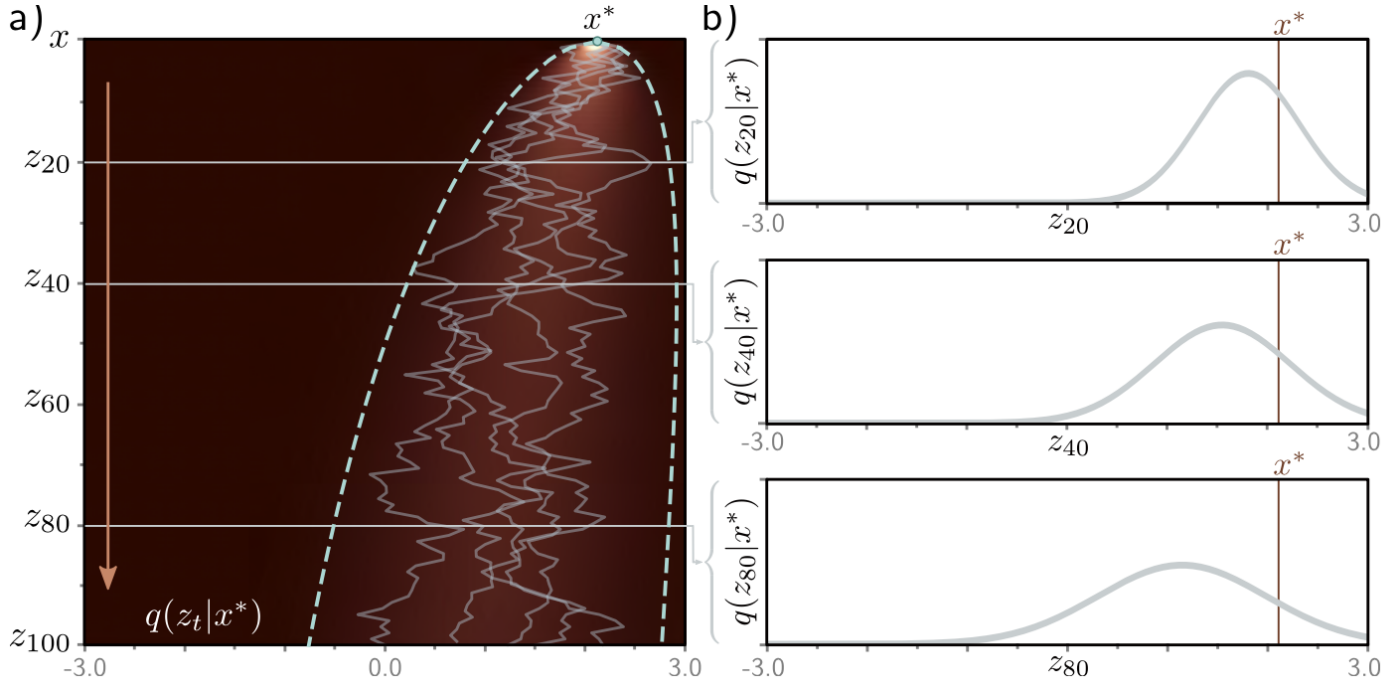


Diffusion Kernel

$$q(Z_t \mid X)$$


$q(z_t|z_{t-1})$
x
z_1
z_2
z_{t-1}
z_t
z_T
$q(z_t|x)$

Diffusion Kernel



adapted from UDL (Prince), Figure 18.3

$$Z_1 = \sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1$$
$$Z_2 = \sqrt{1-\beta_2}\, Z_1 + \sqrt{\beta_2}\, \epsilon_2$$

$$\begin{aligned} Z_2 &= \sqrt{1-\beta_2}\left(\sqrt{1-\beta_1}\, X + \sqrt{\beta_1}\, \epsilon_1\right) + \sqrt{\beta_2}\, \epsilon_2 \\ &= \sqrt{(1-\beta_2)(1-\beta_1)}\, X + \sqrt{\beta_1(1-\beta_2)}\, \epsilon_1 + \sqrt{\beta_2}\, \epsilon_2 \\ &\approx \sqrt{(1-\beta_2)(1-\beta_1)}\, X + \sqrt{1-(1-\beta_1)(1-\beta_2)}\, \epsilon \end{aligned}$$

Introducing $\alpha_t = \prod_{s=1}^{t}(1-\beta_s)$

$$Z_t = \sqrt{\alpha_t}\, X + \sqrt{1-\alpha_t}\, \epsilon$$

Thus

$$q(Z_t \mid X) = N(\sqrt{\alpha_t}\, X,\, (1-\alpha_t) I)$$

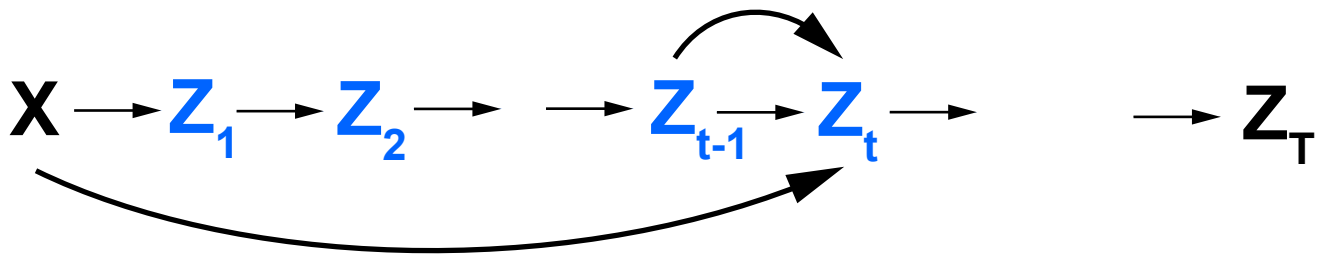
Diffusion Forward

ENCODER

data → noise

forward/diffusion process

$$q(z_t|z_{t-1}) = N(z_t; \sqrt{1-ß_t} z_{t-1}, ß_t I)$$



$$q(z_t|x) = N(z_t; \sqrt{\alpha_t}\, x, (1-\alpha_t) I) \qquad \alpha_t = \prod_{s=1}^{t}(1-ß_s)$$

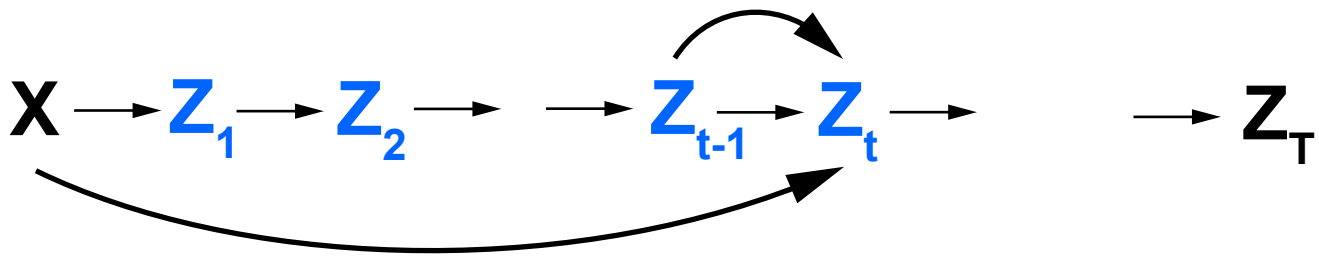
Diffusion Forward

ENCODER

data → noise

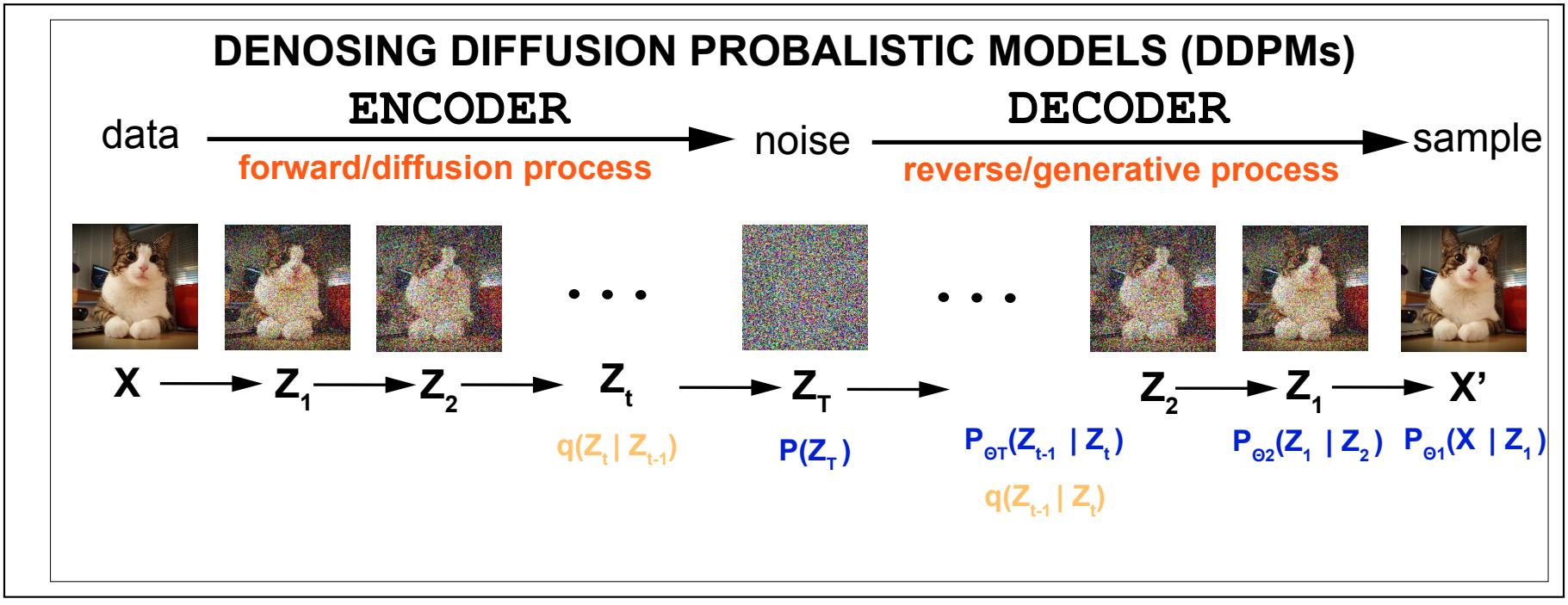
forward/diffusion process

$$q(z_t|z_{t-1}) = N(z_t; \sqrt{1-ß_t} z_{t-1}, ß_t I)$$



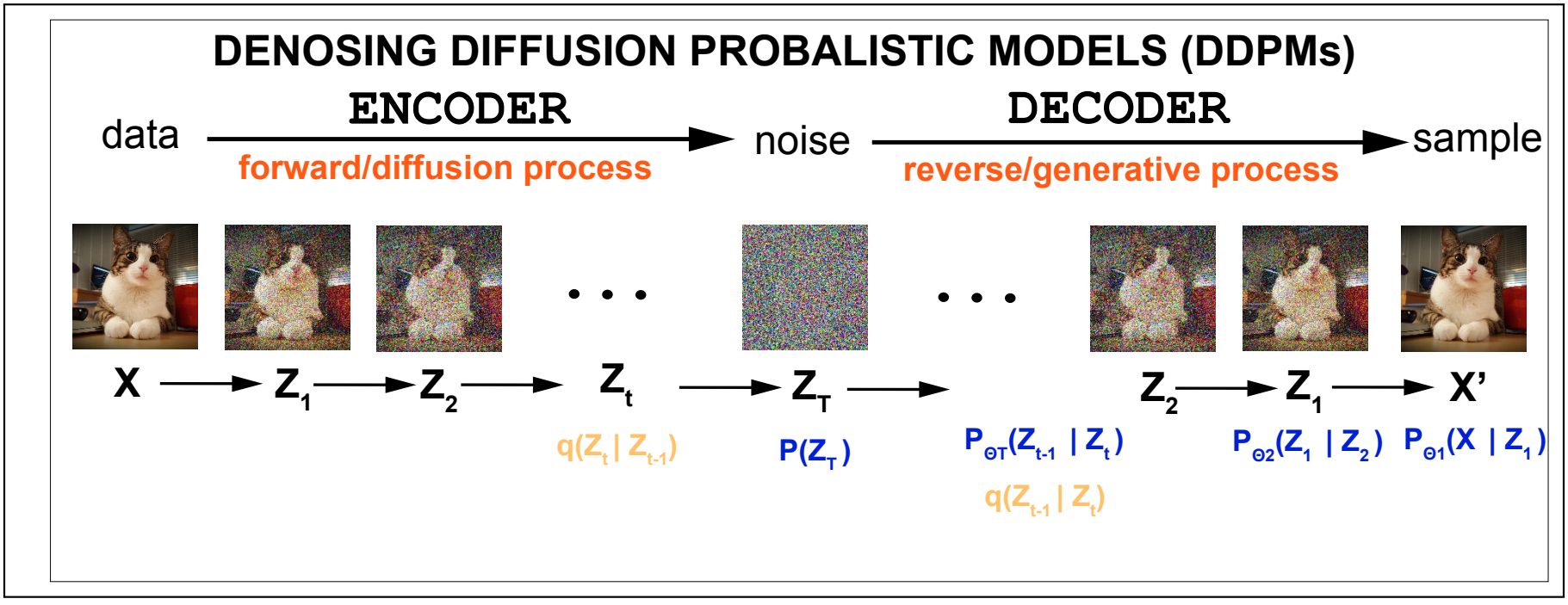
$$q(z_t|x) = N(z_t; \sqrt{\alpha_t}\, x, (1-\alpha_t) I) \qquad \alpha_t = \prod_{s=1}^{t}(1-ß_s)$$

The Intractable Posteriors



$$q(Z_{t-1} \mid Z_t)$$

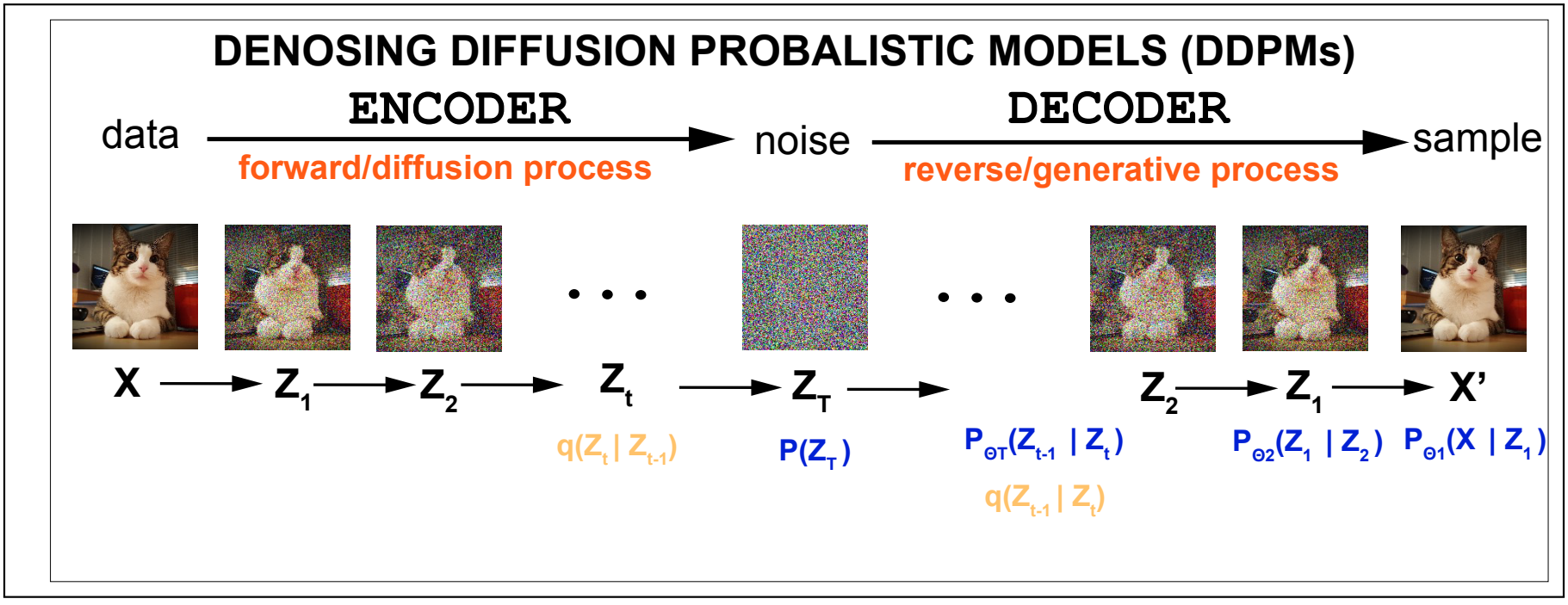
The Intractable Posteriors



$$q(Z_{t-1} \mid Z_t)$$

$$q(Z_{t-1} \mid Z_t) = \frac{q(Z_t \mid Z_{t-1})\, q(Z_{t-1})}{q(Z_t)}$$

The Intractable Posteriors



$$q(Z_{t-1} \mid Z_t)$$

$$q(Z_{t-1} \mid Z_t) = \frac{q(Z_t \mid Z_{t-1})\, q(Z_{t-1})}{q(Z_t)}$$

$$q(Z_t) \ ?$$

The Intractable Posteriors $q(Z_t)$?

The Intractable Posteriors $q(Z_t)$?

$$q(Z_t) = \int_x dX \int_{z_1} dZ_1 \cdots \int_{z_{t-1}} dZ_{t-1} \quad q(Z_1 \ldots Z_t \mid X)\, P(X)$$

The Intractable Posteriors $q(Z_t)$?

$$q(Z_t) = \int_x dX \int_{z_1} dZ_1 \cdots \int_{z_{t-1}} dZ_{t-1} \quad q(Z_1 \ldots Z_t \mid X)\, P(X)$$

$$q(Z_t) = \int_x dX \quad q(Z_t \mid X)\, P(X)$$

The Intractable Posteriors $q(Z_t)$?

$$q(Z_t) = \int_x dX \int_{z_1} dZ_1 \cdots \int_{z_{t-1}} dZ_{t-1} \quad q(Z_1 \ldots Z_t \mid X)\, P(X)$$

$$q(Z_t) = \textstyle\int_x dX \quad q(Z_t \mid X)\, P(X)$$

but... $P(X)$ is unknown

The Intractable Posteriors $q(Z_t)$?

$$q(Z_t) = \int_x dX \int_{z_1} dZ_1 \cdots \int_{z_{t-1}} dZ_{t-1} \quad q(Z_1 \ldots Z_t \mid X)\, P(X)$$

$$q(Z_t) = \int_x dX \quad q(Z_t \mid X)\, P(X)$$

but... $P(X)$ is unknown

Approximated by the Decoder using Normal distributions

The tractable Posteriors

$$q(Z_{t-1} \mid Z_t\, X)$$

$$\begin{aligned} q(Z_{t-1} \mid Z_t\, X) &= \frac{q(Z_t \mid Z_{t-1}\, X)\, q(Z_{t-1} \mid X)}{q(Z_t \mid X)} \\ &\propto q(Z_t \mid Z_{t-1})\, q(Z_{t-1} \mid X) \\ &= N(Z_t; \sqrt{1-\beta_t} Z_{t-1}, \beta_t I)\, N(Z_{t-1}; \sqrt{1 = \alpha_{t-1}} X, (1-\alpha_{t-1} \end{aligned}$$

$$q(Z_{t-1} \mid Z_t\, X) = N(Z_{t-1}; \mu(Z_t, X))$$

The Decoder NNs

DECODER

noise → sample

reverse/generative process

$Z_T \rightarrow \cdots \rightarrow Z_t \rightarrow Z_{t-1} \rightarrow \cdots \rightarrow Z_2 \rightarrow Z_1 \rightarrow X$

$P(z_T) = N(z_T ; 0, I)$

$P_\Theta(z_{t-1} | z_t) = N(z_{t-1} ; \mu_\Theta(z_t,t), \sigma_t I)$

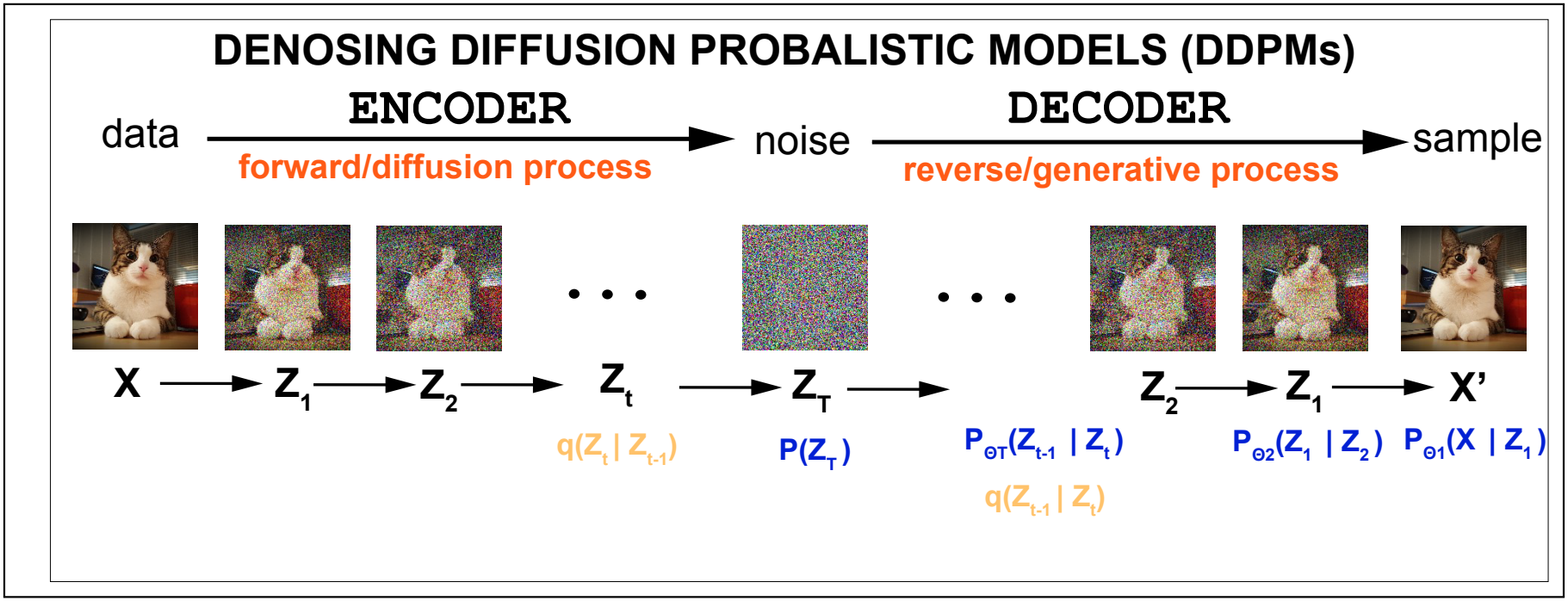
Normal approximation to $q(z_{t-1} | z_t)$, ok if β_t is small

Each $\mu_{\theta_t}(Z_t)$

is a Neural Network with parameters θ_t

σ_t are predetermined

Decoder



$$P(X, Z_1, \ldots Z_T; \theta_1 \ldots \theta_T) = P(Z_T) \prod_{t=2}^{T} P_{\theta_t}(Z_{t-1} \mid Z_t)\, P_{\theta_1}(X \mid Z_1)$$

The Marginal probability of the data:

$$P(X; \theta_1 \ldots \theta_T) = \int dZ_{1:T} P(Z_T) \prod_{t=2}^{T} P_{\theta_t}(Z_{t-1} \mid Z_t)\, P_{\theta_1}(X \mid Z_1)$$

DDPM Training

Training Objective (Unsupervised)

$$P_{data}(X) \approx P(X; \theta_1 \dots \theta_T)$$

Equivalent to

$$\theta^*_{1:T} = \text{argmax}_{\theta_{1:T}} \log P(X; \theta_1 \dots \theta_T)$$

DDPM Training

Training Objective (Unsupervised)

$$P_{data}(X) \approx P(X; \theta_1 \dots \theta_T)$$

Equivalent to

$$\theta^*_{1:T} = \text{argmax}_{\theta_{1:T}} \log P(X; \theta_1 \dots \theta_T)$$

The DDPM Loss
Evidence Lower Bound

$$\begin{aligned}
\log P_{\theta_{1:T}}(X) &= \log \left[\int P_{\theta_{1:T}}(X, Z_{1:T})\, dZ_{1:T} \right] \\
&= \log \left[\int q(Z_{1:T} \mid X)\, \frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)}\, dZ_{1:T} \right] \\
&\geq \int q(Z_{1:T} \mid X)\, \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}
\end{aligned}$$

$$ELBO(\theta_{1:T}) = \int q(Z_{1:T} \mid X)\, \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}$$

The DDPM Loss
Evidence Lower Bound

$$\log P_{\theta_{1:T}}(X) = \log \left[\int P_{\theta_{1:T}}(X, Z_{1:T})\, dZ_{1:T} \right]$$

$$= \log \left[\int q(Z_{1:T} \mid X) \frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)}\, dZ_{1:T} \right]$$

$$\geq \int q(Z_{1:T} \mid X) \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}$$

$$ELBO(\theta_{1:T}) = \int q(Z_{1:T} \mid X) \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}$$

$$ELBO(\theta_{1:T}) = \int q(Z_{1:T} \mid X) \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}$$

The distributions are known

$$q(Z_t \mid Z_{t-1}) = N(\sqrt{1-\beta_t}\, Z_{t-1}, \beta_t I)$$
$$P_{\theta_t}(Z_{t-1} \mid Z_t) = N(\mu_{\theta_t}(Z_t), \sigma_t^2 I)$$

Use parameterization

$$\mu_{\theta_t}(Z_t) = \frac{1}{\sqrt{1-\beta_t}}\, Z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\, \epsilon_{\theta_t}(Z_t)$$

Then

$$Loss = -ELBO(\theta_{1:T}) = \sum_{i=1}^{I} \sum_{t=1}^{T} ||\epsilon_{\theta_t}[\sqrt{1-\alpha_t} X_i + \sqrt{\alpha_t}\epsilon_{it}] - \epsilon_{it}||^2$$

$$ELBO(\theta_{1:T}) = \int q(Z_{1:T} \mid X) \log \left[\frac{P_{\theta_{1:T}}(X, Z_{1:T})}{q(Z_{1:T} \mid X)} \right] dZ_{1:T}$$

The distributions are known

$$q(Z_t \mid Z_{t-1}) = N(\sqrt{1-\beta_t}\, Z_{t-1}, \beta_t I)$$
$$P_{\theta_t}(Z_{t-1} \mid Z_t) = N(\mu_{\theta_t}(Z_t), \sigma_t^2 I)$$

Use parameterization

$$\mu_{\theta_t}(Z_t) = \frac{1}{\sqrt{1-\beta_t}} Z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}} \epsilon_{\theta_t}(Z_t)$$

NN for mean **NN for error**

Then

$$Loss = -ELBO(\theta_{1:T}) = \sum_{i=1}^{I} \sum_{t=1}^{T} ||\epsilon_{\theta_t}[\underbrace{\sqrt{1-\alpha_t} X_i + \sqrt{\alpha_t}\epsilon_{it}}_{Z_t}] - \epsilon_{it}||^2$$

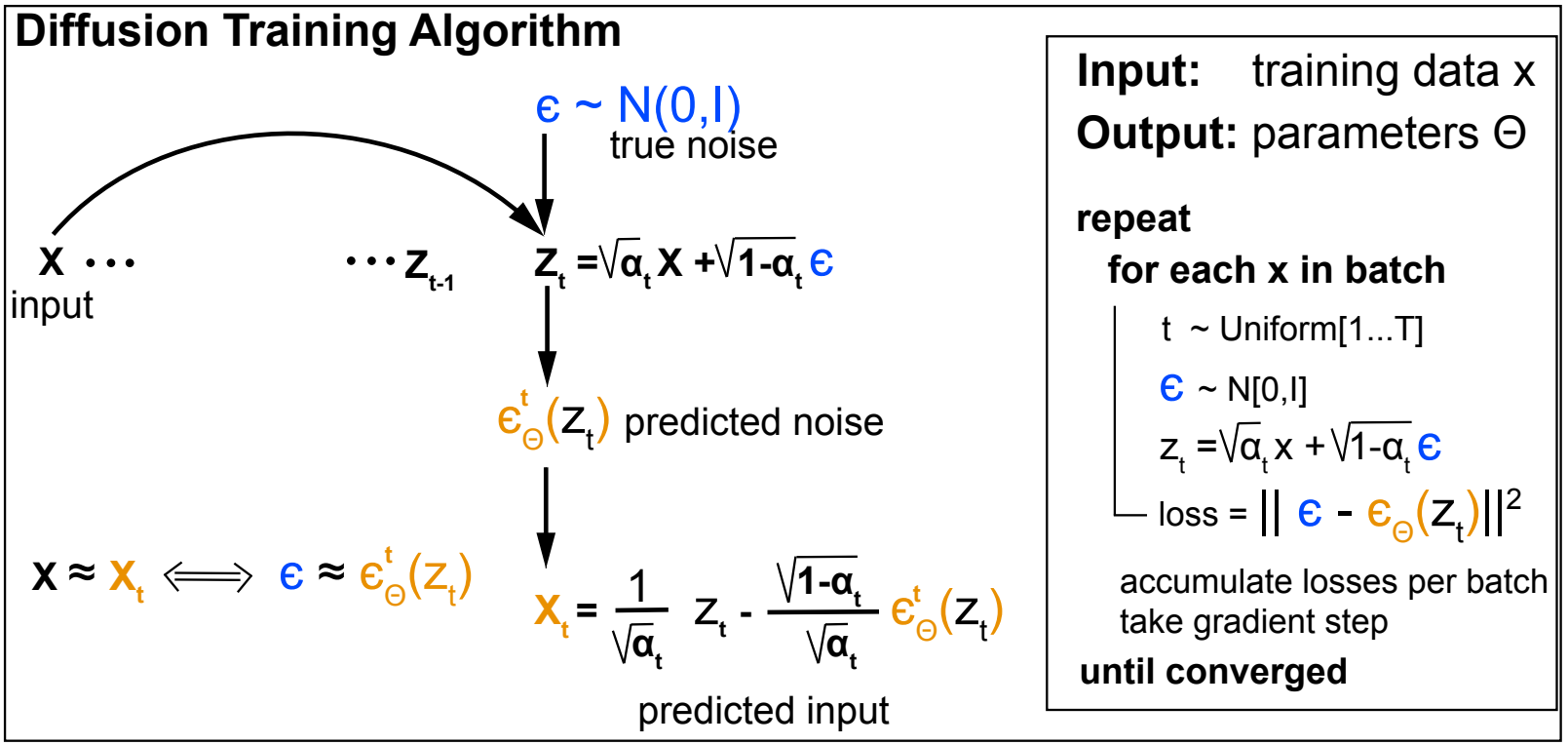
$$Loss = -ELBO(\theta_{1:T}) = \sum_{i=1}^{I} \sum_{t=1}^{T} ||\epsilon_{\theta_t}[\sqrt{1-\alpha_t}X_i + \sqrt{\alpha_t}\epsilon_{it}] - \epsilon_{it}||^2$$

$$Loss = -ELBO(\theta_{1:T}) = \sum_{i=1}^{I} \sum_{t=1}^{T} || \epsilon_{\theta_t} [\sqrt{1-\alpha_t} X_i + \sqrt{\alpha_t} \epsilon_{it}] - \epsilon_{it} ||^2$$

error calculated by the decoder

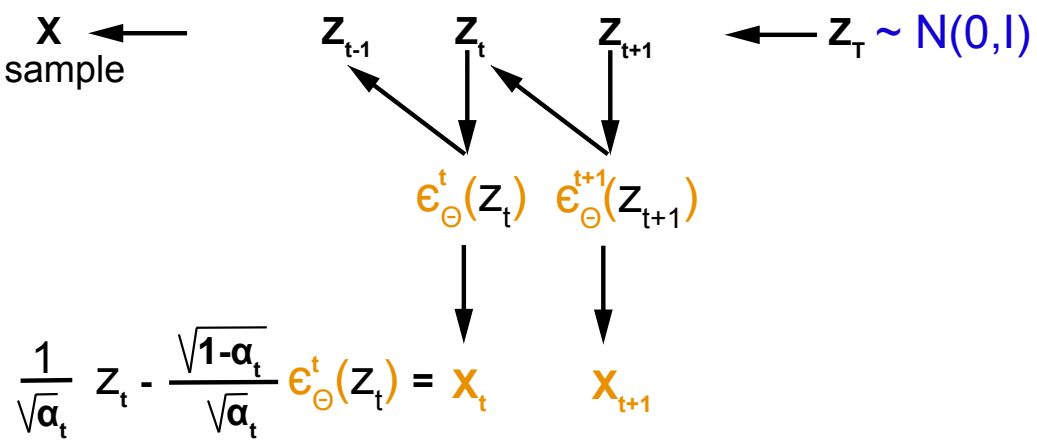
error applied by the encoder

DDPM Training Algorithm



DDPM Sampling Algorithm

Diffusion Sampling Algorithm



Input: modes $\epsilon_\Theta^{1:T}(\ )$

Output: sample x

$z_T \sim N(0,I)$

for t in [T..2] **do**

$z_{t-1} = \frac{1}{\sqrt{1-\beta_t}} z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}} \epsilon_\Theta^t(z_t)$

$\epsilon \sim N(0,I)$

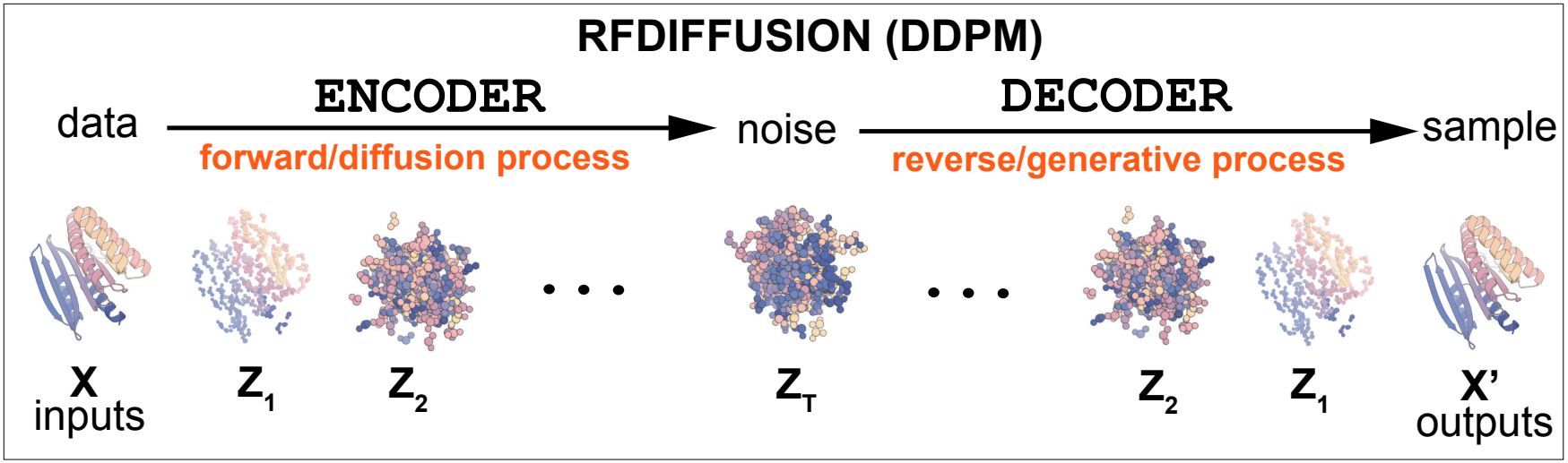
$z_{t-1} \mathrel{+}= \sigma_t \epsilon$

$x = \frac{1}{\sqrt{1-\beta_1}} z_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}} \epsilon_\Theta^1(z_1)$

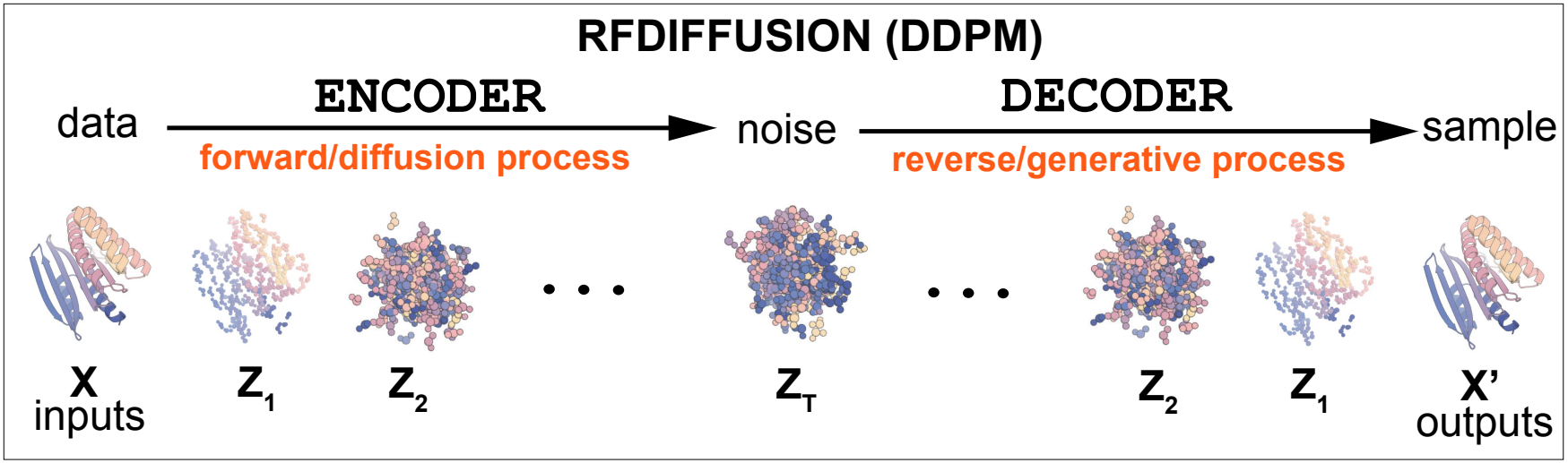
$$P_{\theta_t}(Z_{t-1} \mid Z_t) = N(\mu_{\theta_t}(Z_t), \sigma_t^2 I)$$

$$\mu_{\theta_t}(Z_t) = \frac{1}{\sqrt{1-\beta_t}} Z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}} \epsilon_{\theta_t}(Z_t)$$

De novo design of protein structure

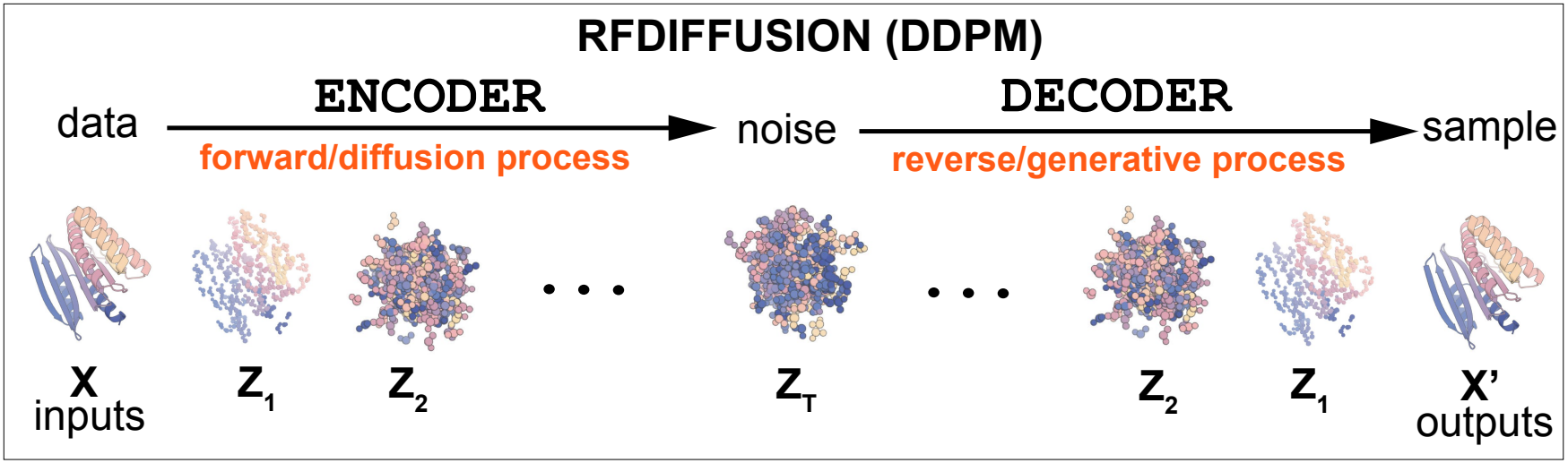


De novo design of protein structure



The input is a real 3D protein structure

De novo design of protein structure

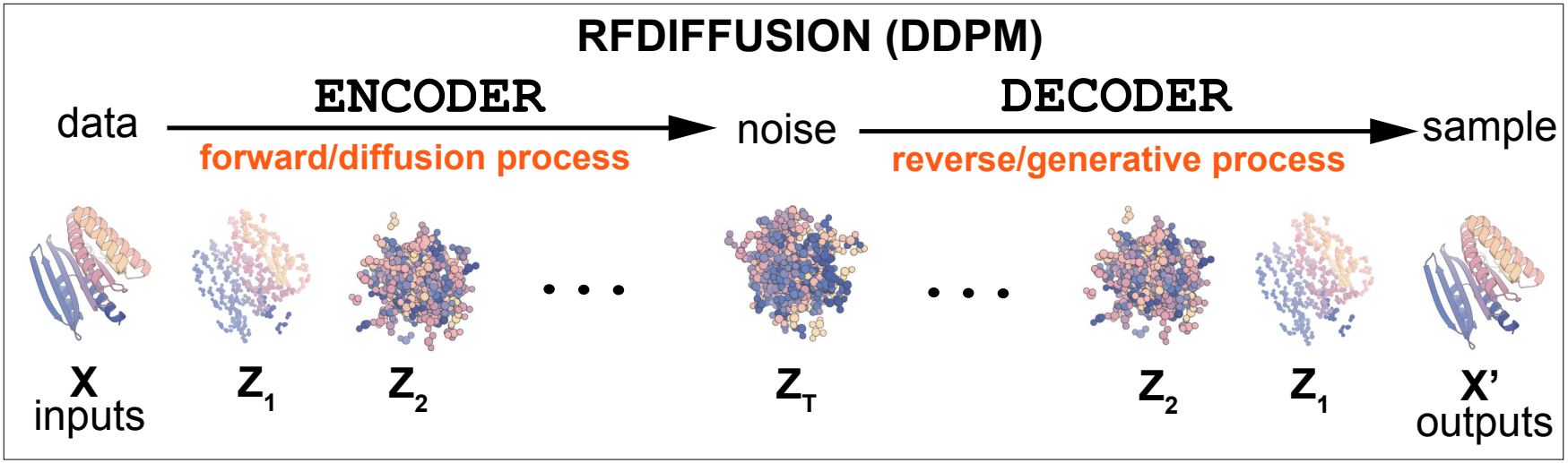


The input is a real 3D protein structure

The noise is applied to the reference frames of each aa

$T_i = (r_i, t_i)$ rotation + translations [SU3]

De novo design of protein structure



The input is a real 3D protein structure

The noise is applied to the reference frames of each aa

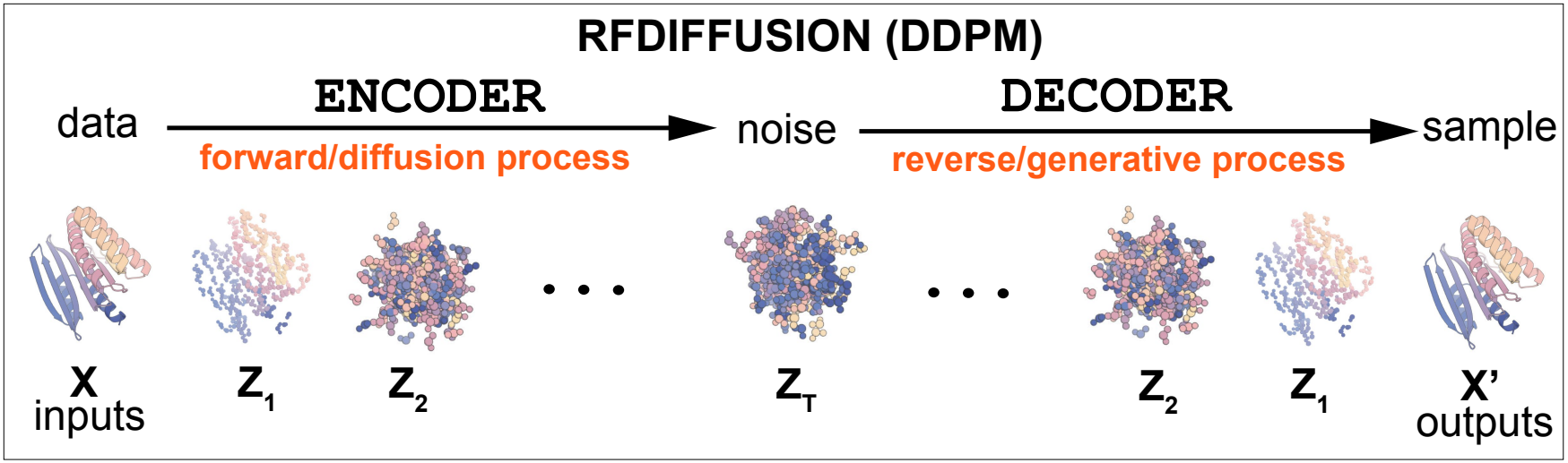
$T_i = (r_i, t_i)$ rotation + translations [SU3]

The **Encoder** works on two variables

$Z = (t_1, \ldots, t_L)$

$r = (r_1, \ldots, r_L)$

De novo design of protein structure



The input is a real 3D protein structure

The noise is applied to the reference frames of each aa

$T_i = (r_i, t_i)$ rotation + translations [SU3]

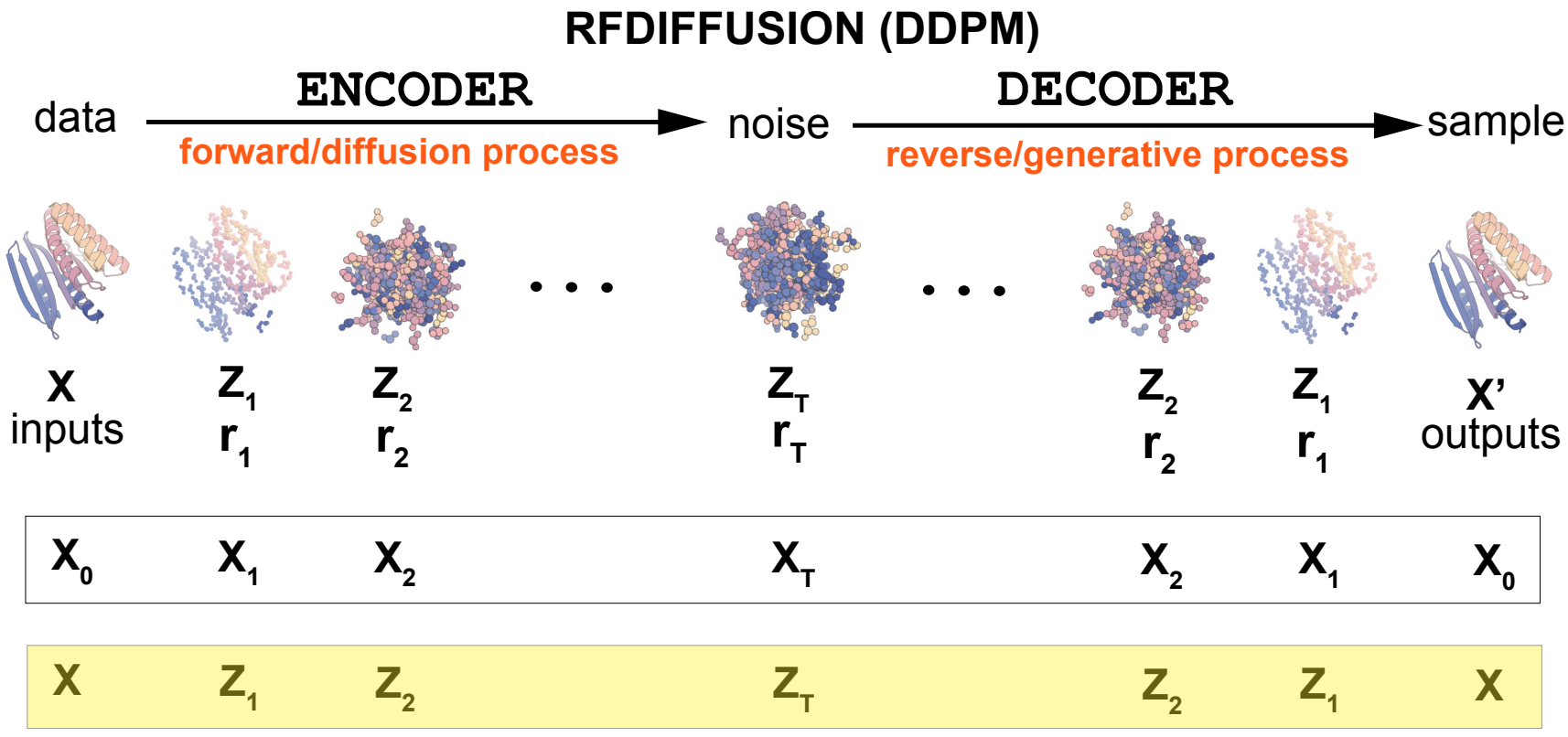
The **Encoder** works on two variables

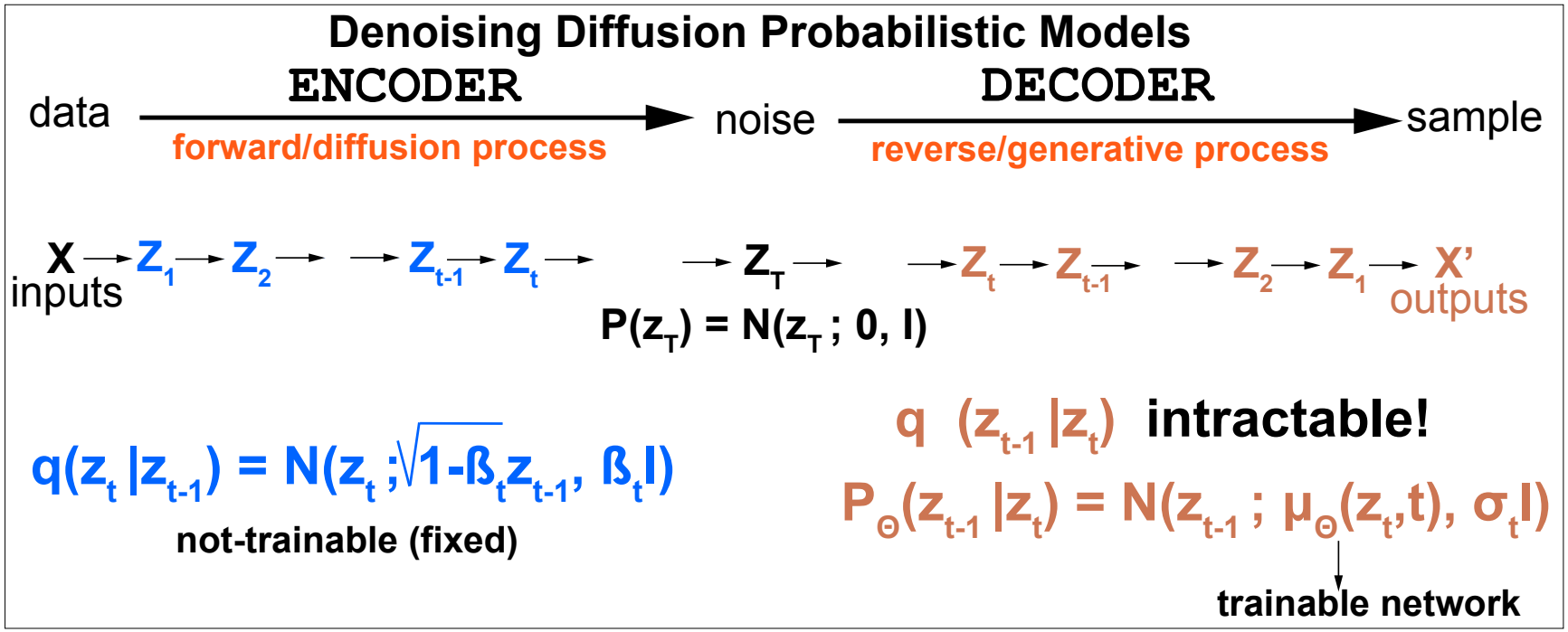
$Z = (t_1, \ldots, t_L)$

$r = (r_1, \ldots, r_L)$

The **Decoder** is a variant of **RoseTTAFold** (RF)

De novo design of protein structure




Denoising Diffusion Probabilistic Models
data
ENCODER
forward/diffusion process
noise
DECODER
reverse/generative process
sample
X
inputs
Z_1 Z_2 Z_{t-1} Z_t Z_T Z_t Z_{t-1} Z_2 Z_1 X'
outputs
$P(z_T) = N(z_T ; 0, I)$
$q(z_t | z_{t-1}) = N(z_t ; \sqrt{1-ß_t} z_{t-1}, ß_t I)$
not-trainable (fixed)
$q(z_{t-1} | z_t)$ intractable!
$P_\Theta(z_{t-1} | z_t) = N(z_{t-1} ; \mu_\Theta(z_t,t), \sigma_t I)$
trainable network

amino acid chain = sequence of reference frames

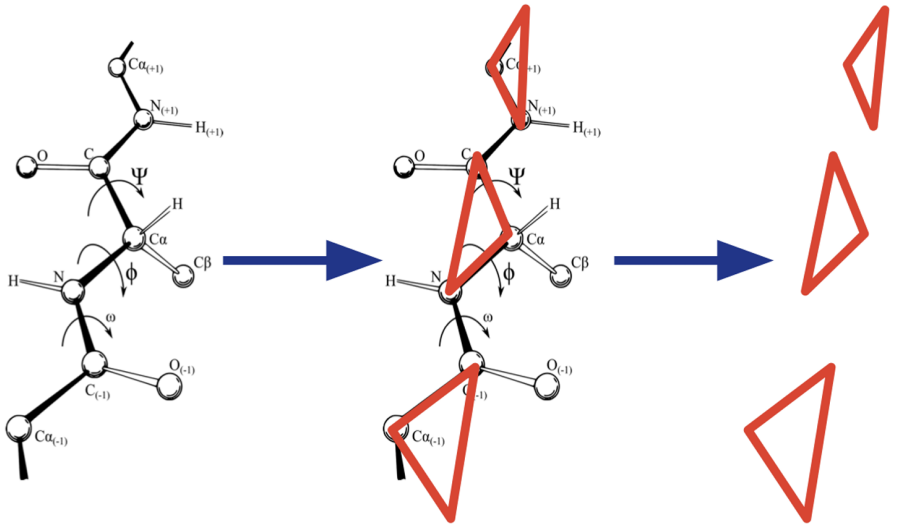
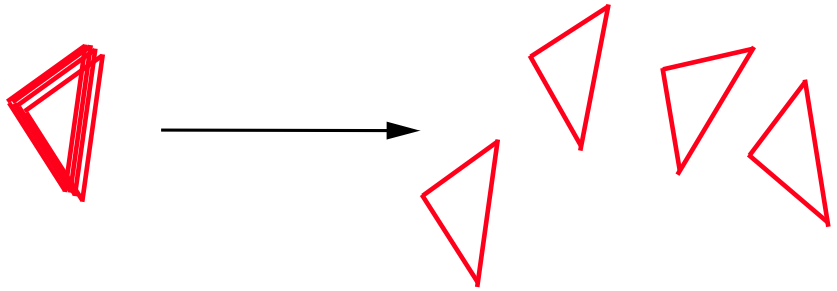


Image: Dcrjsr, vectorised Adam Rędzikowski (CC BY 3.0, Wikipedia)



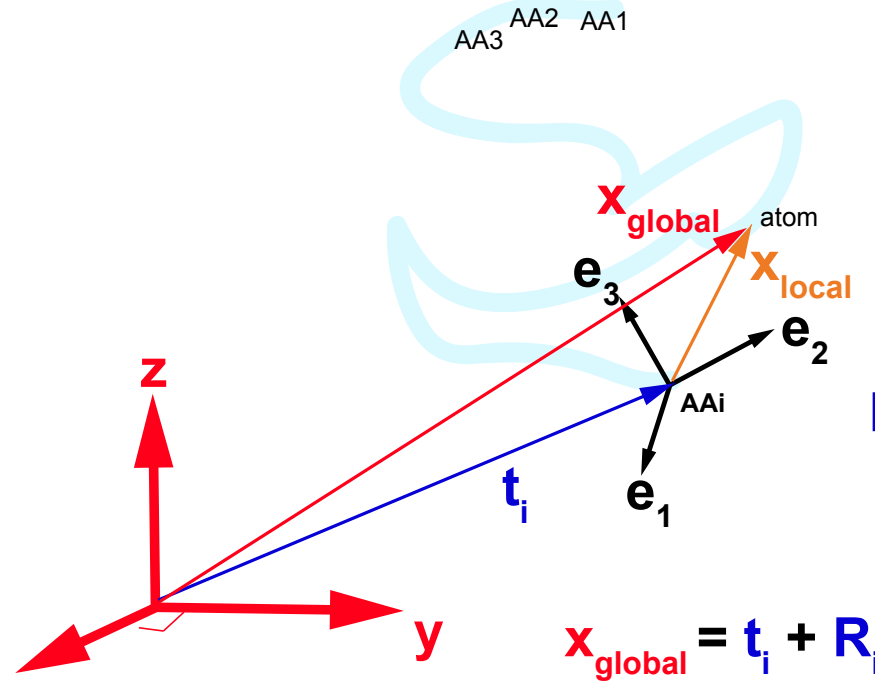
Special Euclean Group

SE(3) = Translation (t) + Rotation (R)

$$t = (t_1, t_2, t_3) \qquad R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

SO(3) = Special Orthogonal group

$$T = (R,t) = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



$$R_i = \text{concat}(e_1, e_2, e_3)$$

$$x_{global} = t_i + R_i\, x_{local} \qquad = T_i\, x_{local}$$

$$x_{local} = R_i^{-1}(x_{global} - t_i) = T_i^{-1}\, x_{global}$$

Denoising Diffusion Probabilistic Models

data —**ENCODER**→ noise —**DECODER**→ sample

forward/diffusion process

reverse/generative process

$X \rightarrow Z_1 \rightarrow Z_2 \rightarrow \cdots \rightarrow Z_{t-1} \rightarrow Z_t \rightarrow \cdots \rightarrow Z_T \rightarrow \cdots \rightarrow Z_t \rightarrow Z_{t-1} \rightarrow \cdots \rightarrow Z_2 \rightarrow Z_1 \rightarrow X'$

inputs … outputs

$P(z_T) = N(z_T ; 0, I)$

$q(z_t | z_{t-1}) = N(z_t ; \sqrt{1-ß_t} z_{t-1}, ß_t I)$

not-trainable (fixed)

$q(z_{t-1} | z_t)$ **intractable!**

$P_\Theta(z_{t-1} | z_t) = N(z_{t-1} ; \mu_\Theta(z_t, t), \sigma_t I)$

trainable network

RFDiffusion

$X \rightarrow Z_1 \rightarrow Z_2 \rightarrow \cdots \rightarrow Z_{t-1} \rightarrow Z_t \rightarrow \cdots \rightarrow Z_T \rightarrow \cdots \rightarrow Z_t \rightarrow Z_{t-1} \rightarrow \cdots \rightarrow Z_2 \rightarrow Z_1 \rightarrow X'$

inputs … outputs

$X_0 \rightarrow X_1 \rightarrow X_2 \rightarrow \cdots \rightarrow X_{t-1} \rightarrow X_t \rightarrow \cdots \rightarrow X_T$

for translations (Z)

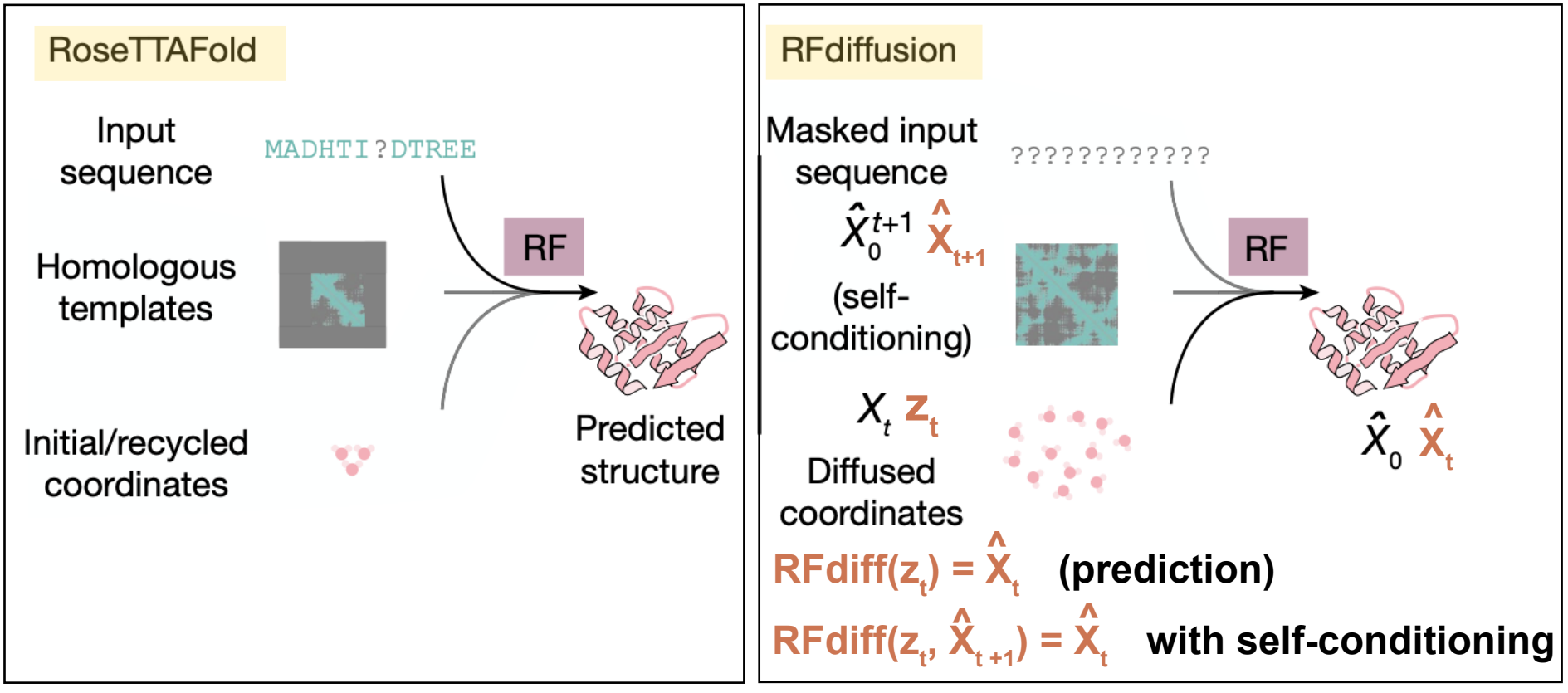
$q(z_t | z_{t-1}) = N(z_t ; 1-ß_t z_{t-1}, ß_t I)$

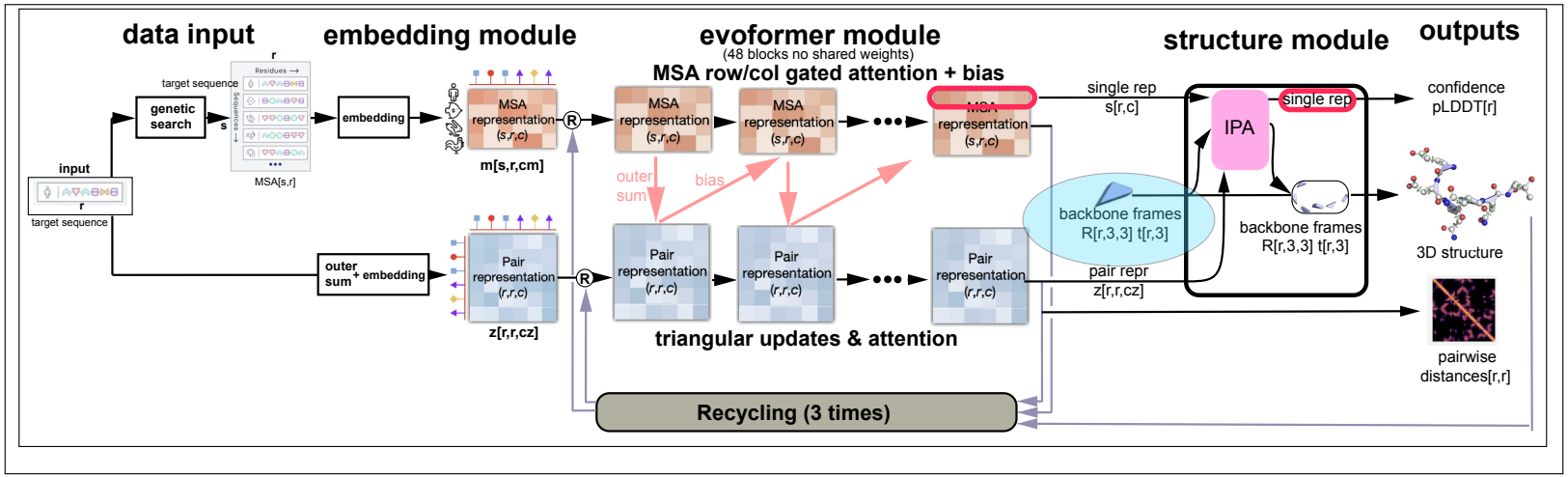
$RFdiff(z_t) = \hat{X}_t$ **(prediction)**

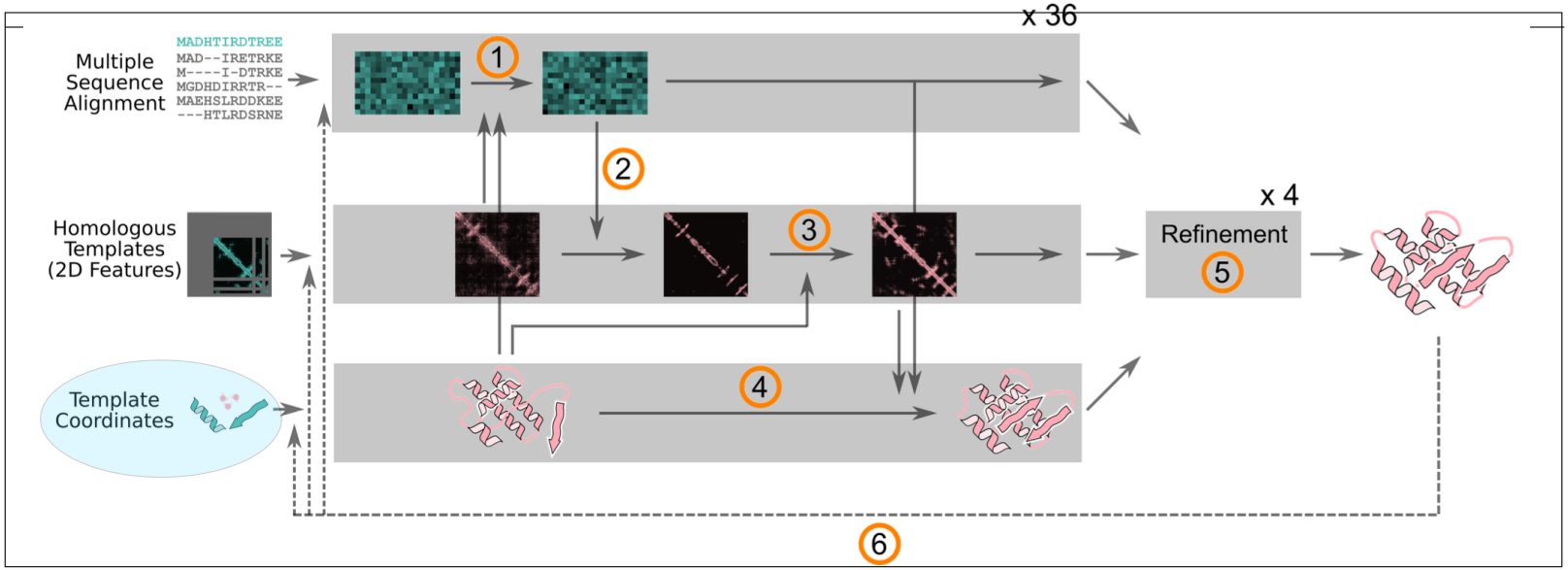
for reference frames (r)

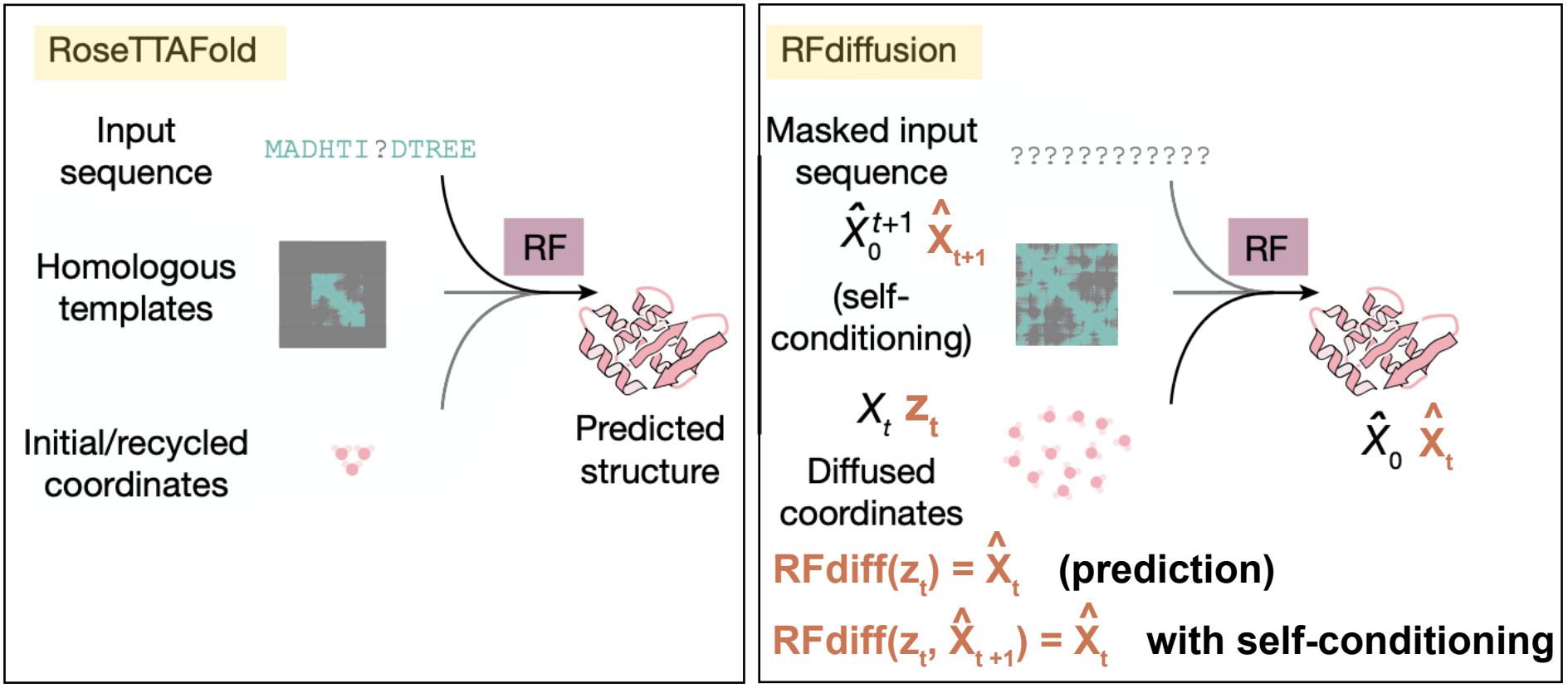
???? SE(3) diffusion

$z_{t-1} = \text{interpol}(z_t, \hat{X}_t)$

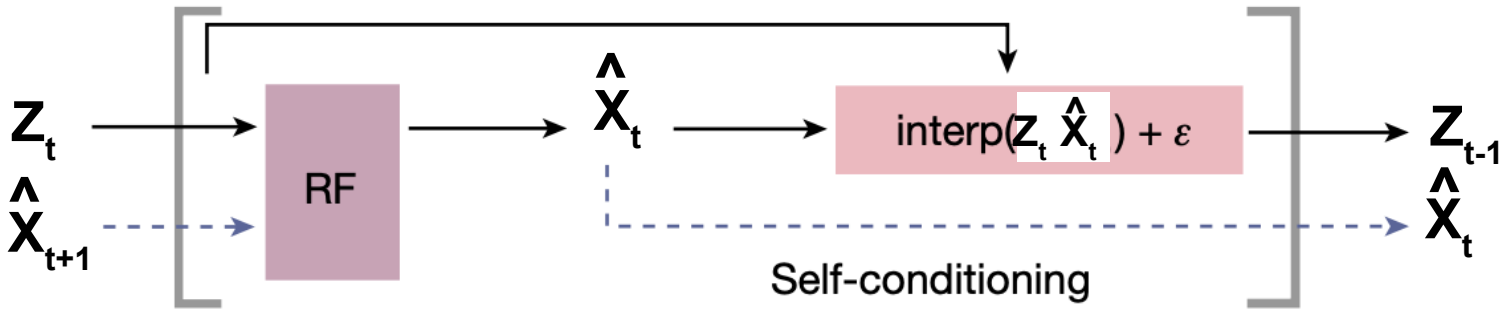

RoseTTAFold
Input sequence
MADHTI?DTREE
Homologous templates
RF
Initial/recycled coordinates
Predicted structure
RFdiffusion
Masked input sequence
??????????
$\hat{X}_0^{t+1}$ $\hat{X}_{t+1}$
(self-conditioning)
X_t z_t
Diffused coordinates
RF
$\hat{X}_0$ $\hat{X}_t$
RFdiff(z_t) = $\hat{X}_t$ (prediction)
RFdiff(z_t, $\hat{X}_{t+1}$) = $\hat{X}_t$ with self-conditioning


data input
embedding module
evoformer module
(48 blocks no shared weights)
MSA row/col gated attention + bias
structure module
outputs
target sequence
genetic search
MSA[s,r]
input
target sequence
embedding
MSA representation (s,r,c)
m[s,r,cm]
outer sum
bias
outer sum + embedding
Pair representation (r,r,c)
z[r,r,cz]
triangular updates & attention
single rep s[r,c]
IPA
single rep
backbone frames R[r,3,3] t[r,3]
pair repr z[r,r,cz]
confidence pLDDT[r]
3D structure
pairwise distances[r,r]
Recycling (3 times)


x 36
Multiple Sequence Alignment
MADHTIRDTREE
MAD--IRETRKE
M----I-DTRKE
MGDHDIRRTR--
MAEHSLRDDKEE
---HTLRDSRNE
Homologous Templates (2D Features)
Template Coordinates
1
2
3
4
x 4
Refinement
5
6

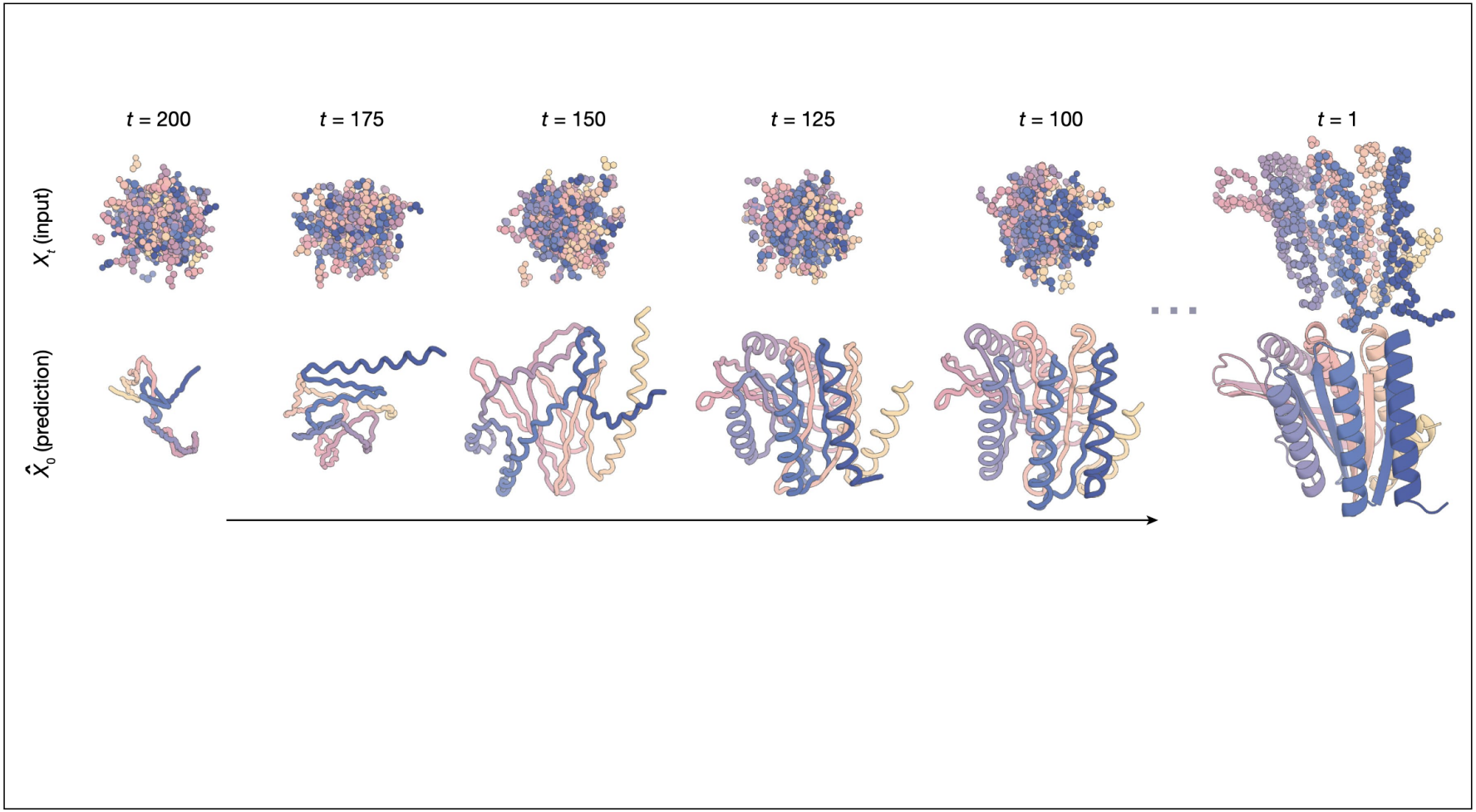

RoseTTAFold
Input sequence
MADHTI?DTREE
Homologous templates
RF
Initial/recycled coordinates
Predicted structure
RFdiffusion
Masked input sequence
??????????
$\hat{X}_0^{t+1}$ $\hat{X}_{t+1}$
(self-conditioning)
X_t z_t
Diffused coordinates
RF
$\hat{X}_0$ $\hat{X}_t$
RFdiff(z_t) = $\hat{X}_t$ (prediction)
RFdiff(z_t, $\hat{X}_{t+1}$) = $\hat{X}_t$ with self-conditioning

Single RFdiffusion step



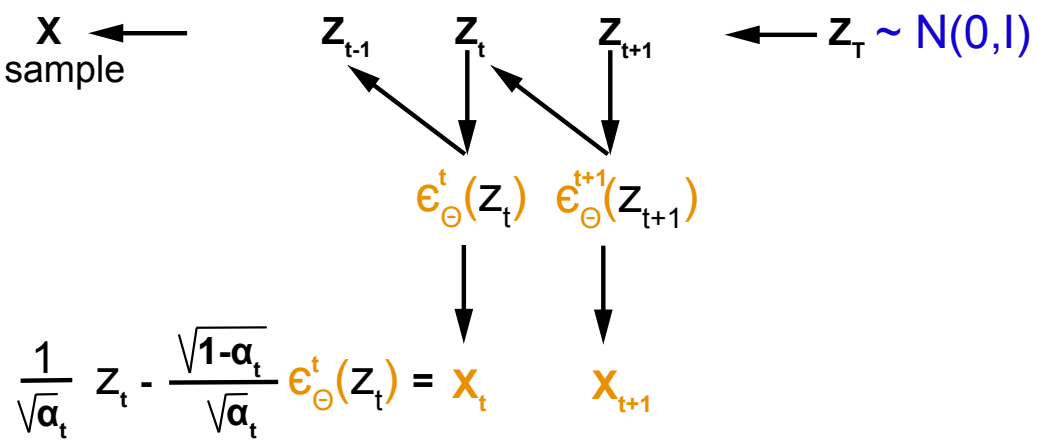
$RFdiff(z_t) = \hat{X}_t$ **(prediction)**

$RFdiff(z_t, \hat{X}_{t+1}) = \hat{X}_t$ **with self-conditioning**


t = 200
t = 175
t = 150
t = 125
t = 100
t = 1
X_t (input)
$\hat{X}_0$ (prediction)

DDPM Sampling Algorithm

Diffusion Sampling Algorithm



Input: modes $\epsilon_\Theta^{1:T}(\ )$

Output: sample x

$z_T \sim N(0,I)$

for t in [T..2] **do**

$z_{t-1} = \frac{1}{\sqrt{1-\beta_t}} z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}} \epsilon_\Theta^t(z_t)$

$\epsilon \sim N(0,I)$

$z_{t-1} \mathrel{+}= \sigma_t \epsilon$

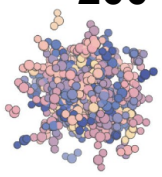
$x = \frac{1}{\sqrt{1-\beta_1}} z_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}} \epsilon_\Theta^1(z_1)$

$$P_{\theta_t}(Z_{t-1} \mid Z_t) = N(\mu_{\theta_t}(Z_t), \sigma_t^2 I)$$

$$\mu_{\theta_t}(Z_t) = \frac{1}{\sqrt{1-\beta_t}} Z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}} \epsilon_{\theta_t}(Z_t)$$

SampleReference(L)

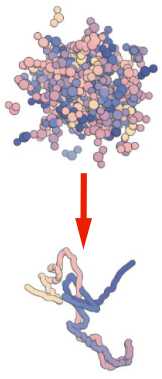
Z_{200}



X_t (input)

$\hat{X}_0$ (prediction)

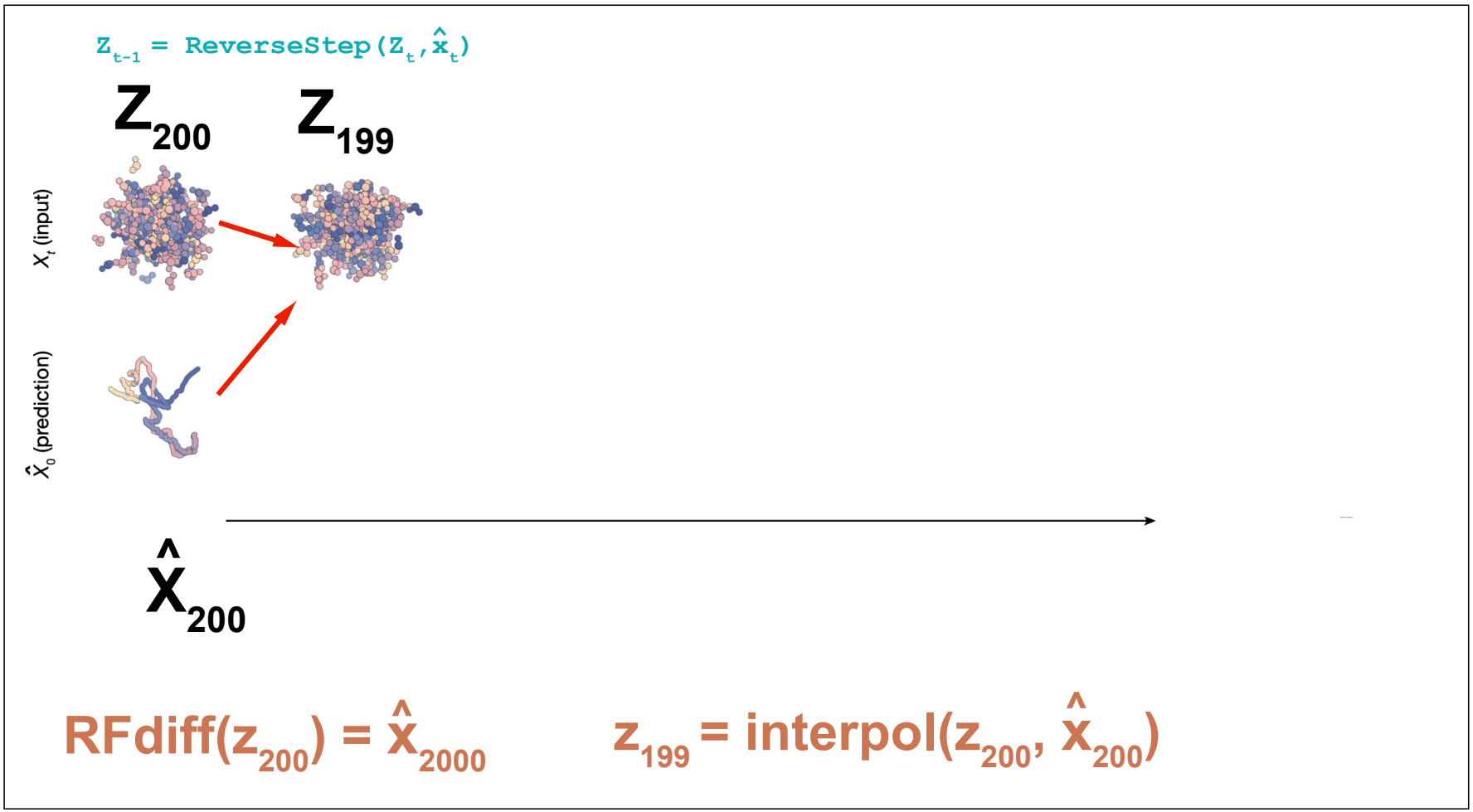
Z_{200}



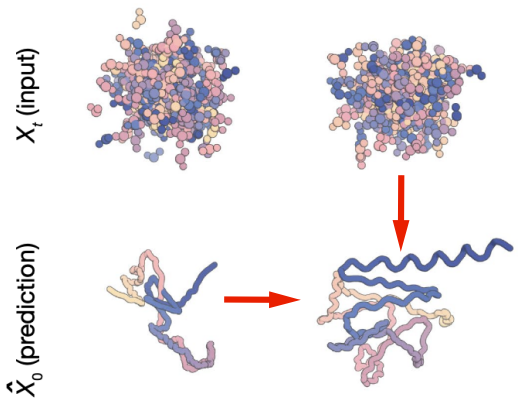
$\hat{X}_{200}$

$\hat{x}_t = \texttt{RFDiffusion}(Z_t)$

$\text{RFdiff}(z_{200}) = \hat{x}_{2000}$


$Z_{t-1} = \text{ReverseStep}(Z_t, \hat{x}_t)$
Z_{200}
Z_{199}
X_t (input)
$\hat{X}_0$ (prediction)
$\hat{X}_{200}$
$\text{RFdiff}(z_{200}) = \hat{x}_{2000}$
$z_{199} = \text{interpol}(z_{200}, \hat{x}_{200})$

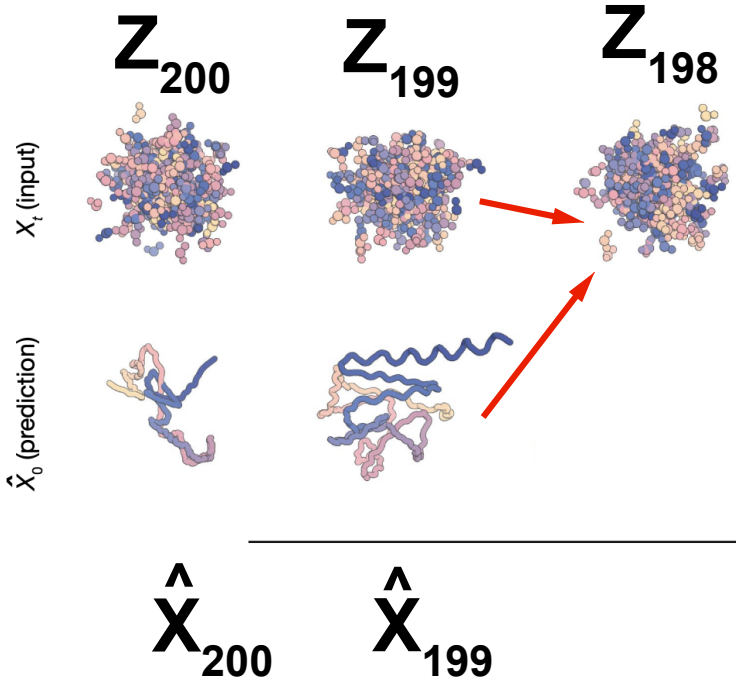
Z_{200} Z_{199}



$\hat{X}_{200}$ $\hat{x}_t = \text{RFDiffusion}(Z_t, \hat{x}_{t+1})$

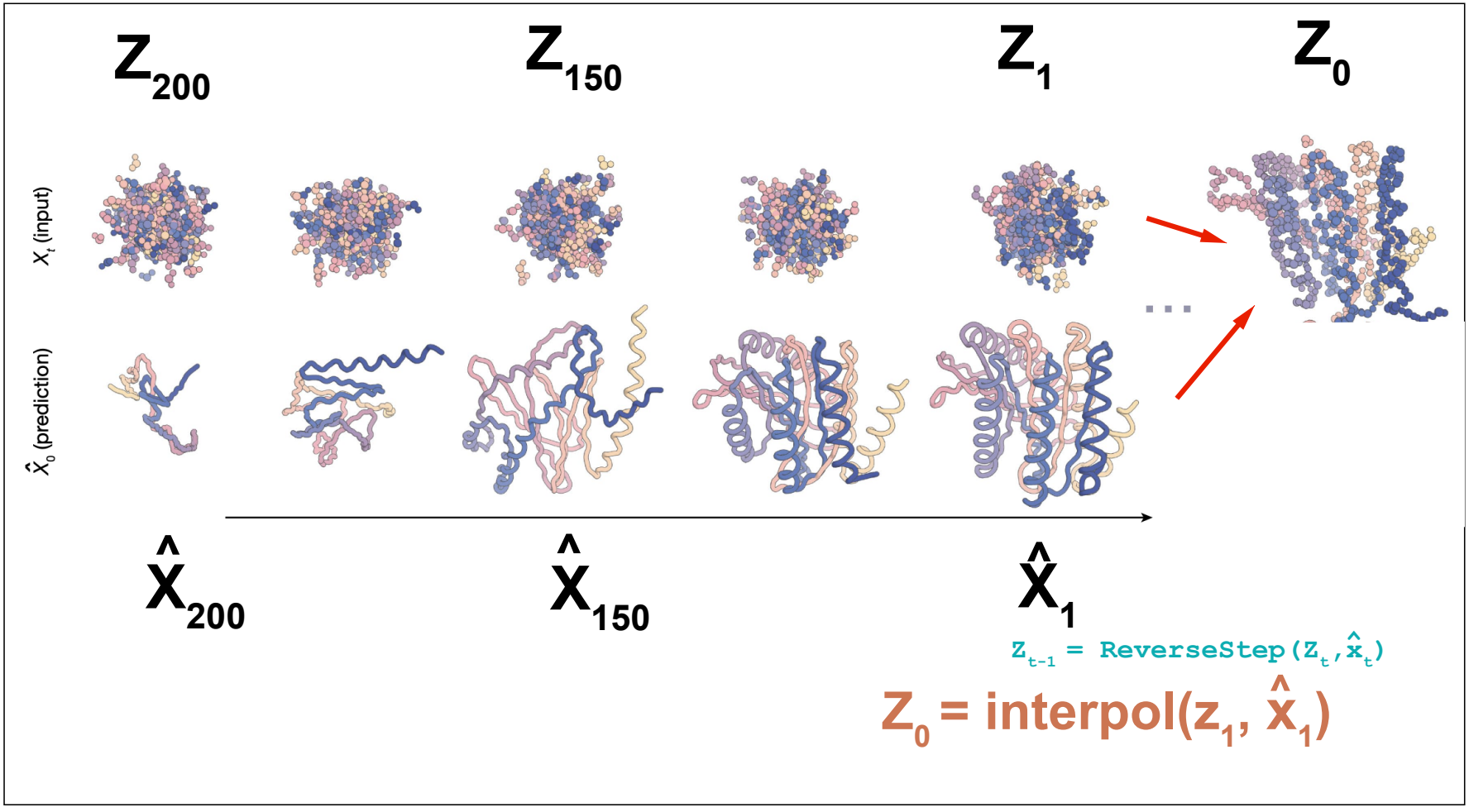
$\text{RFdiff}(z_{199}, \hat{x}_{200}) = \hat{x}_{199}$

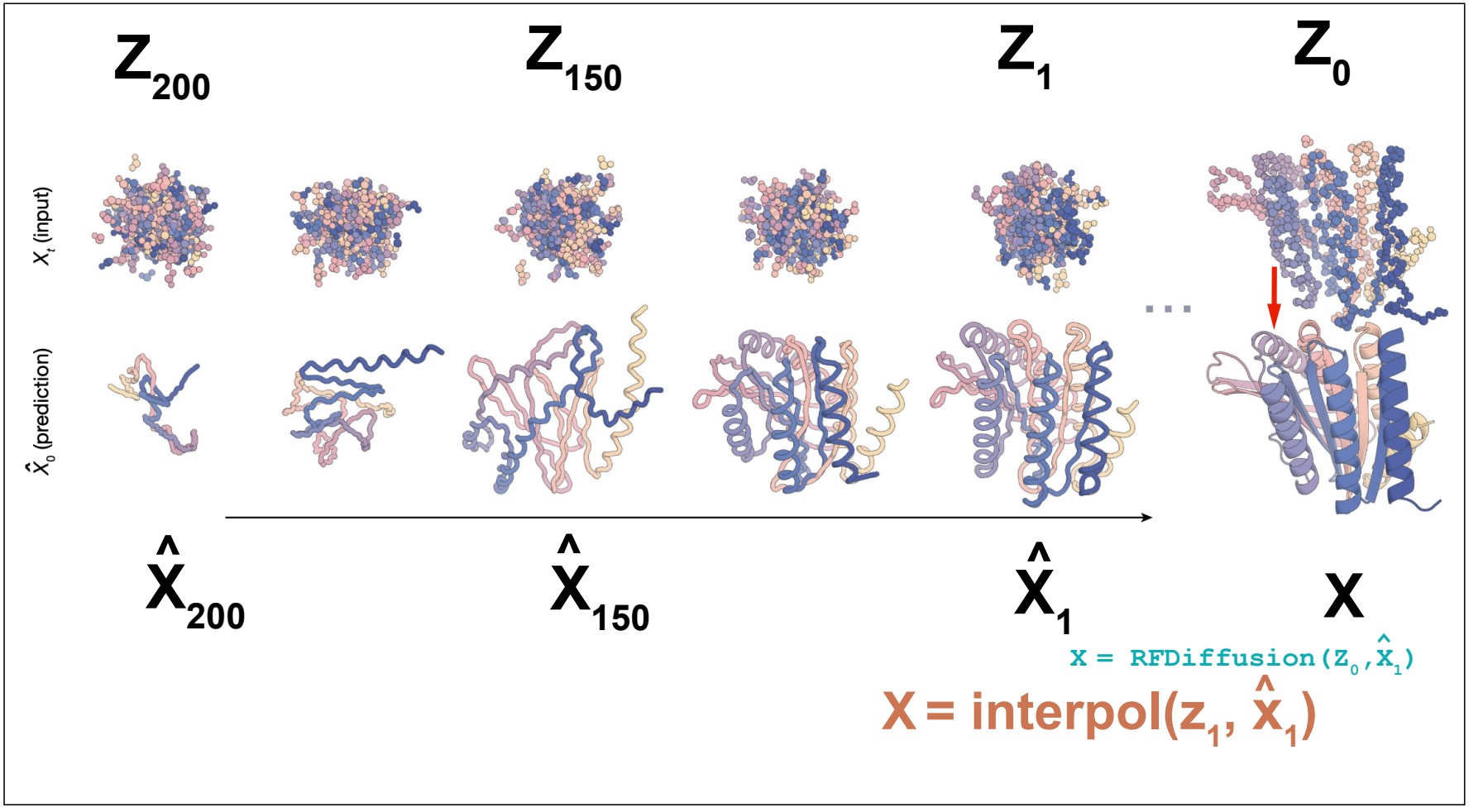
$Z_{t-1} = \text{ReverseStep}(Z_t, \hat{x}_t)$



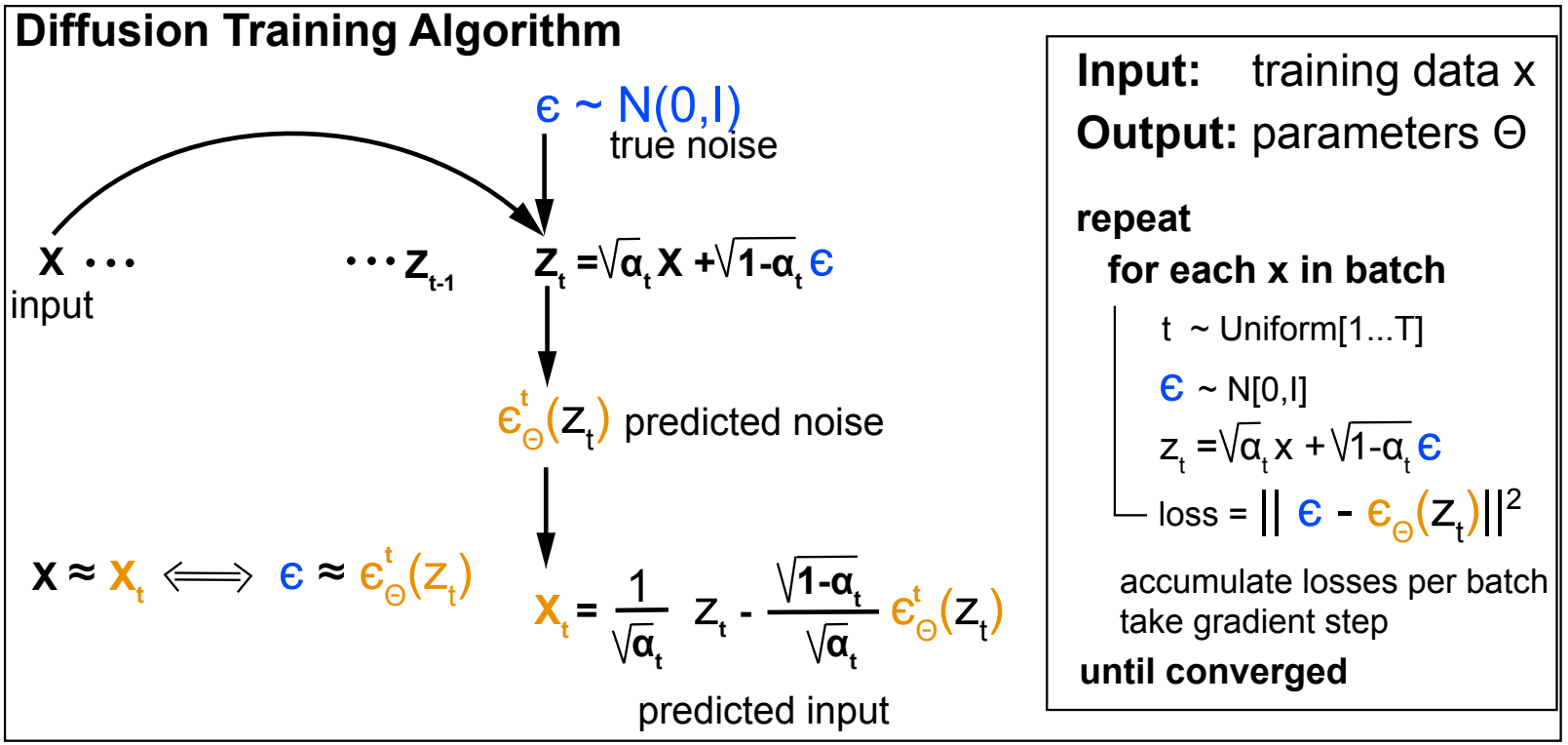
$\text{RFdiff}(z_{199}, \hat{x}_{200}) = \hat{x}_{199}$

$z_{198} = \text{interpol}(z_{199}, \hat{x}_{199})$


z_{200}
z_{150}
z_1
z_0
x_t (input)
$\hat{x}_0$ (prediction)
$\hat{x}_{200}$
$\hat{x}_{150}$
$\hat{x}_1$
$z_{t-1} = \text{ReverseStep}(z_t, \hat{x}_t)$
$z_0 = \text{interpol}(z_1, \hat{x}_1)$

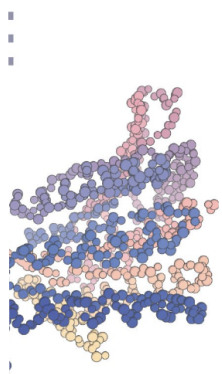

z_{200}
z_{150}
z_1
z_0
x_t (input)
$\hat{x}_0$ (prediction)
$\hat{x}_{200}$
$\hat{x}_{150}$
$\hat{x}_1$
x
$x = \text{RFDiffusion}(z_0, \hat{x}_1)$
$x = \text{interpol}(z_1, \hat{x}_1)$

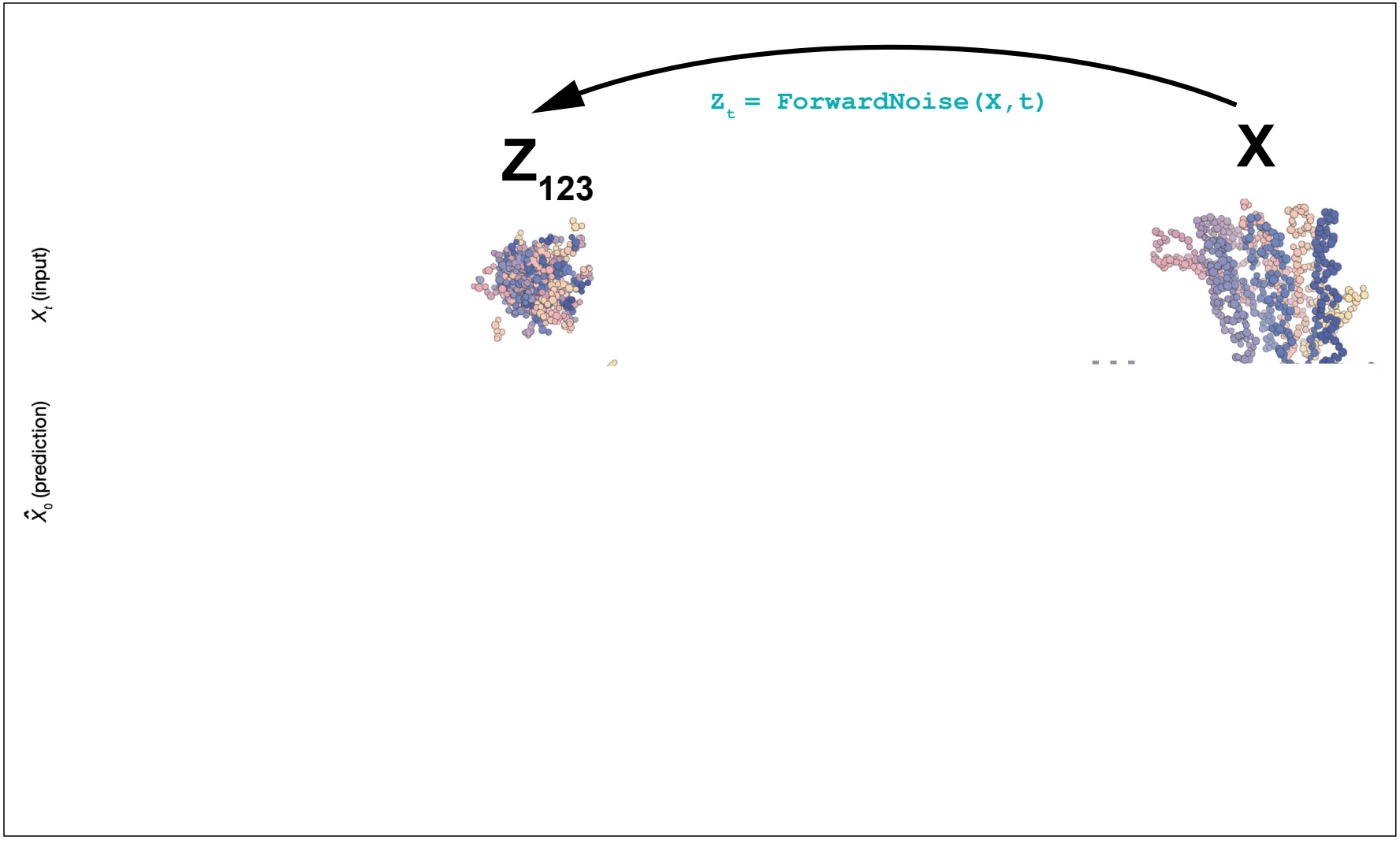
DDPM Training Algorithm

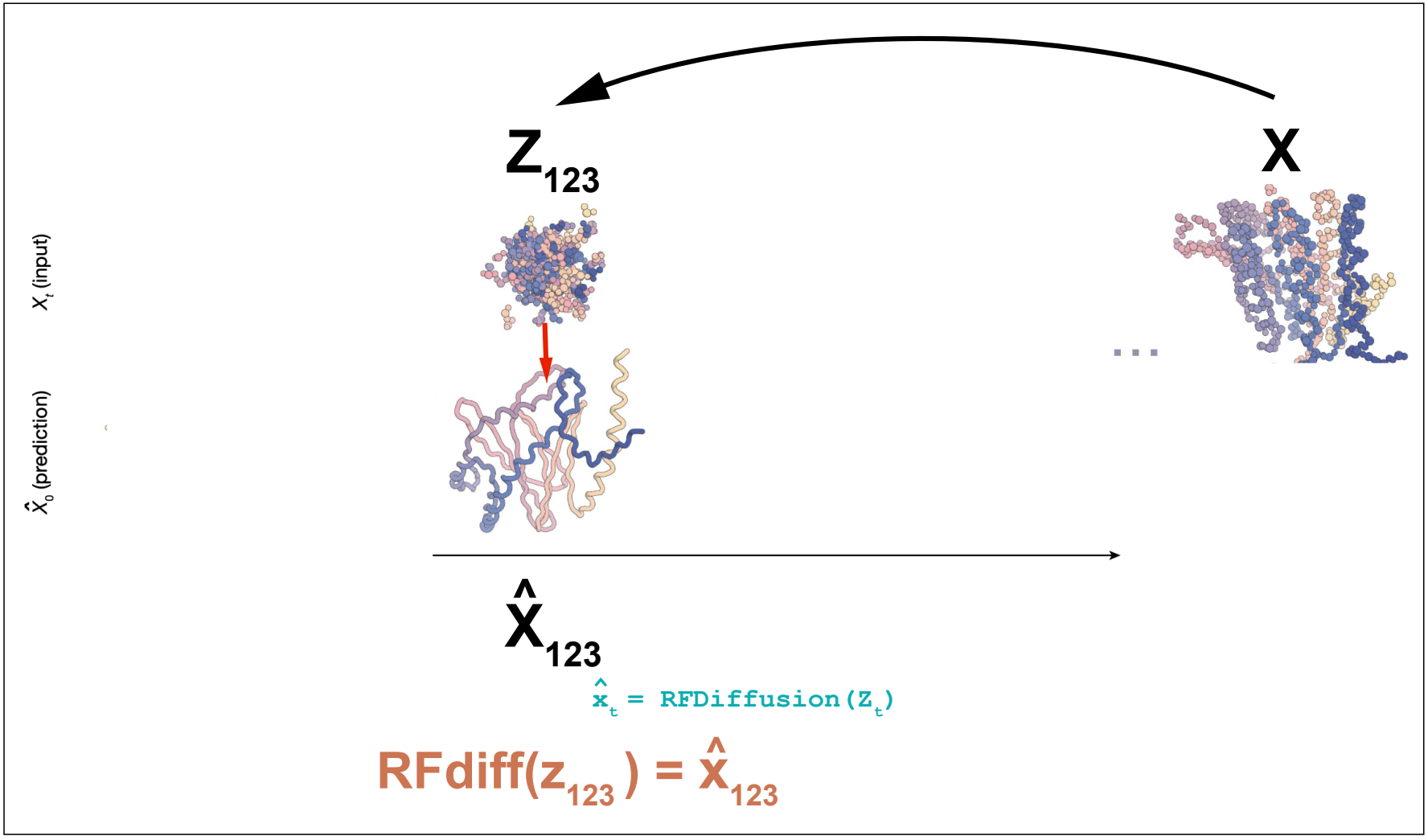


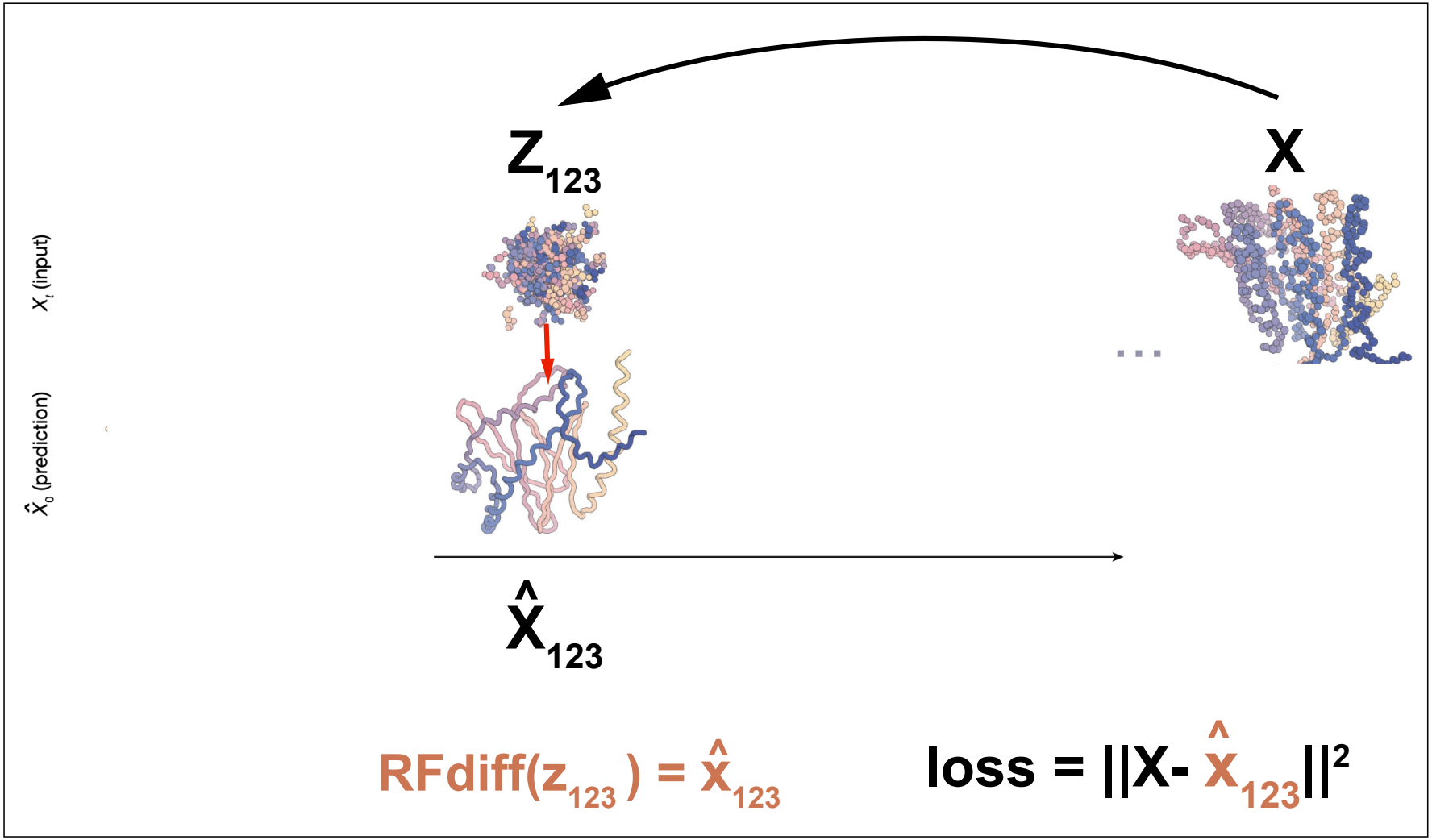
$\hat{X}_0$ (prediction)

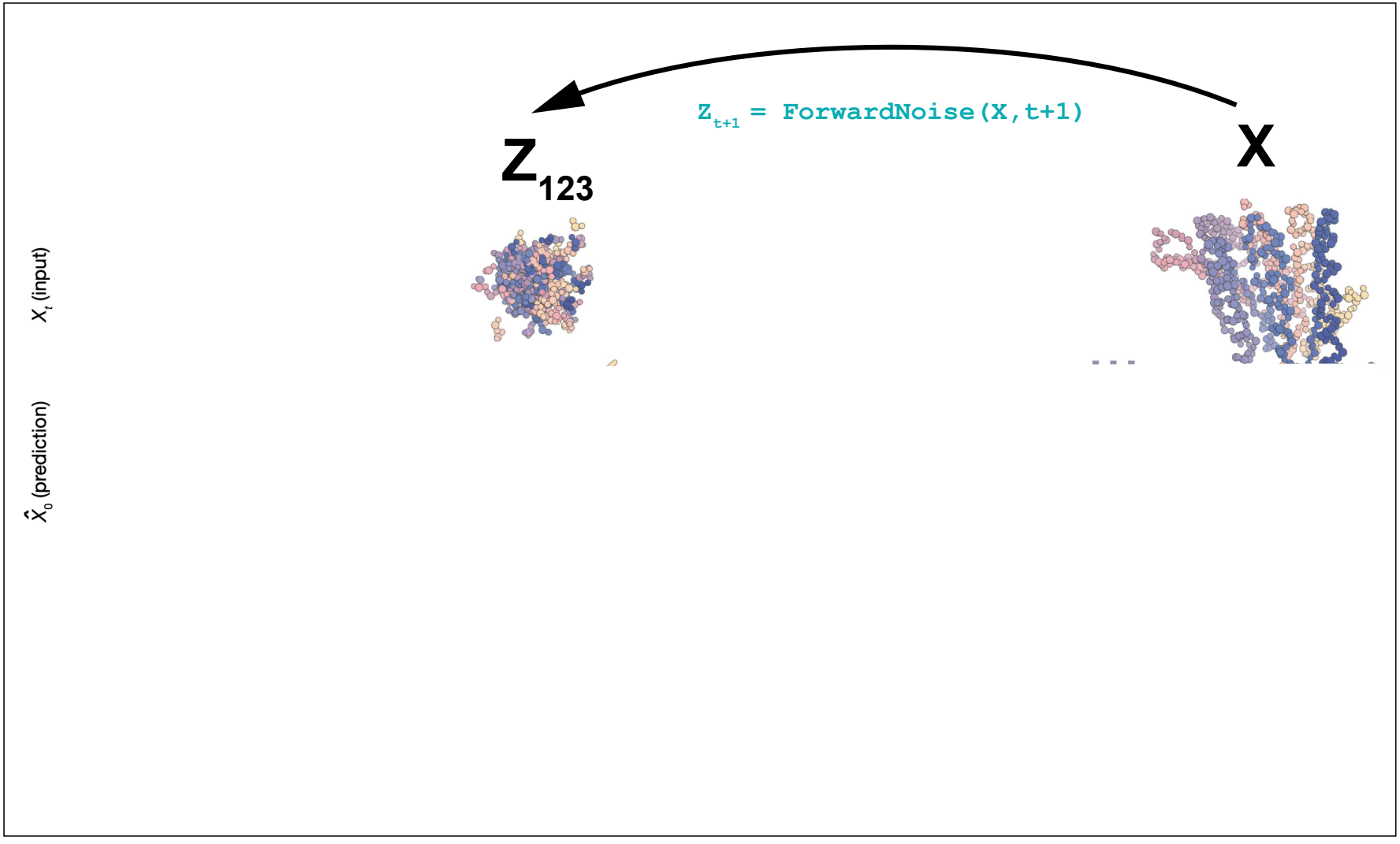
X_t (input)

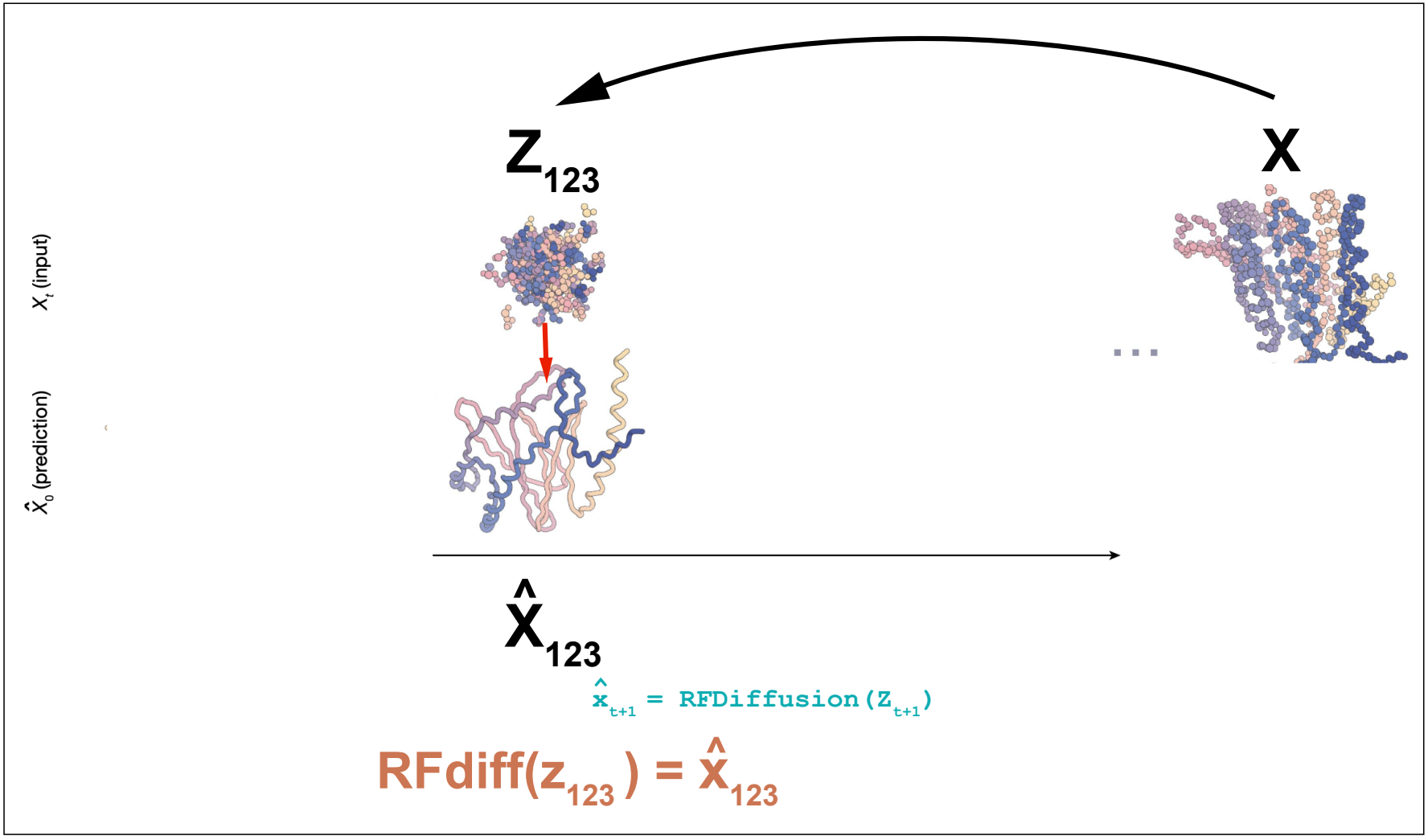


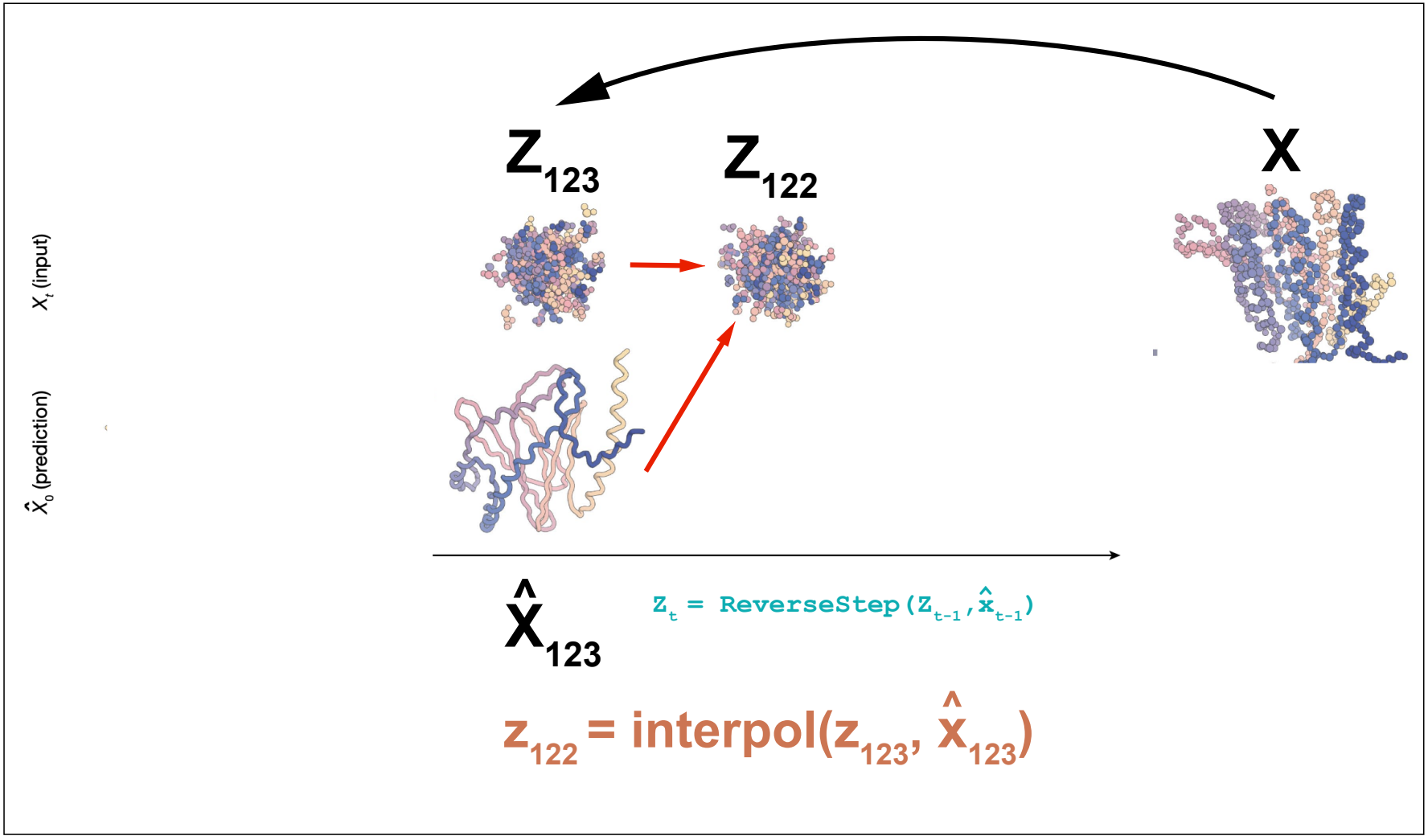

Z_t = ForwardNoise(X,t)
Z_{123}
X
X_t (input)
$\hat{X}_0$ (prediction)

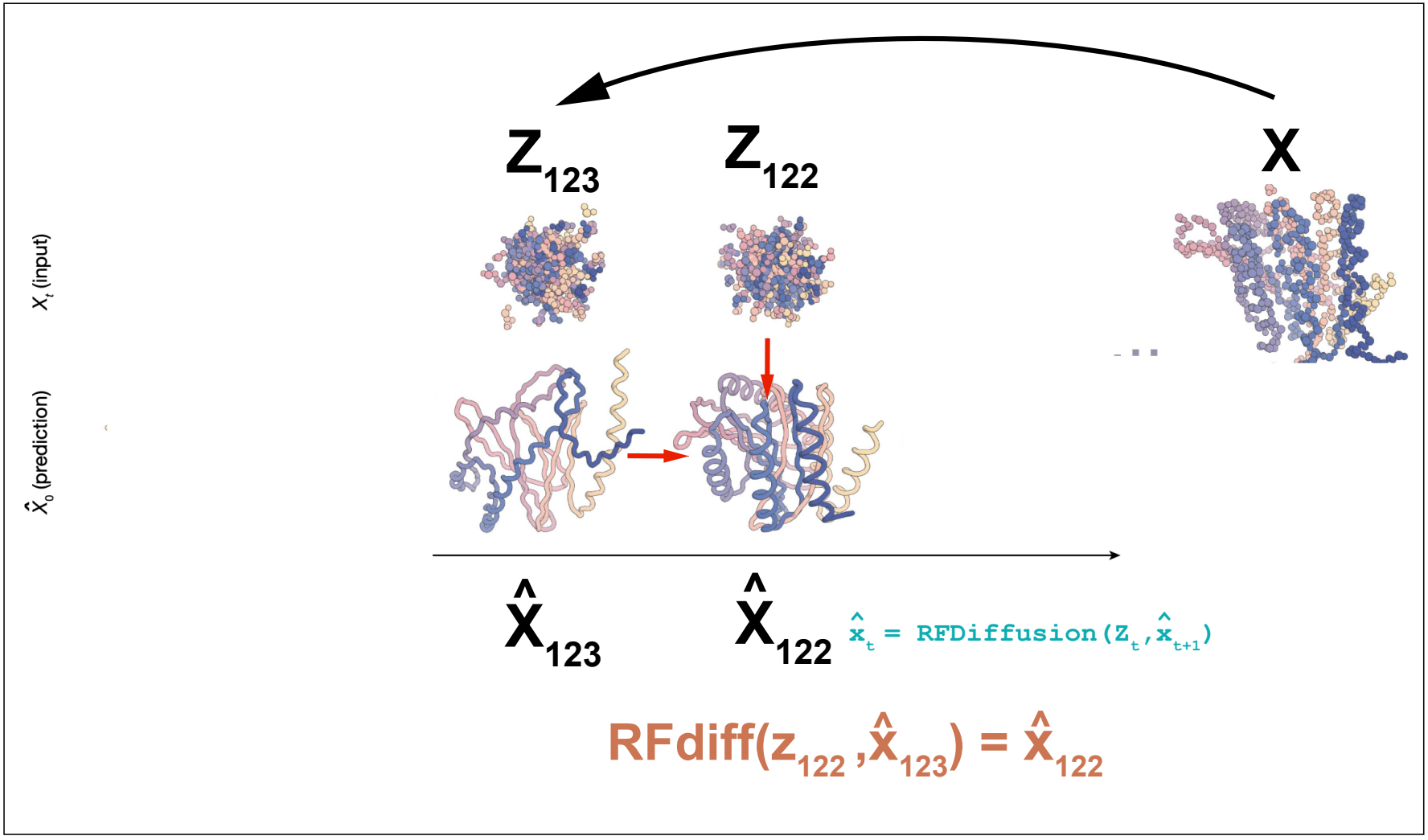

Z_{123}
X
X_t (input)
$\hat{X}_0$ (prediction)
...
$\hat{X}_{123}$
$\hat{x}_t = \text{RFDiffusion}(Z_t)$
$\text{RFdiff}(z_{123}) = \hat{x}_{123}$

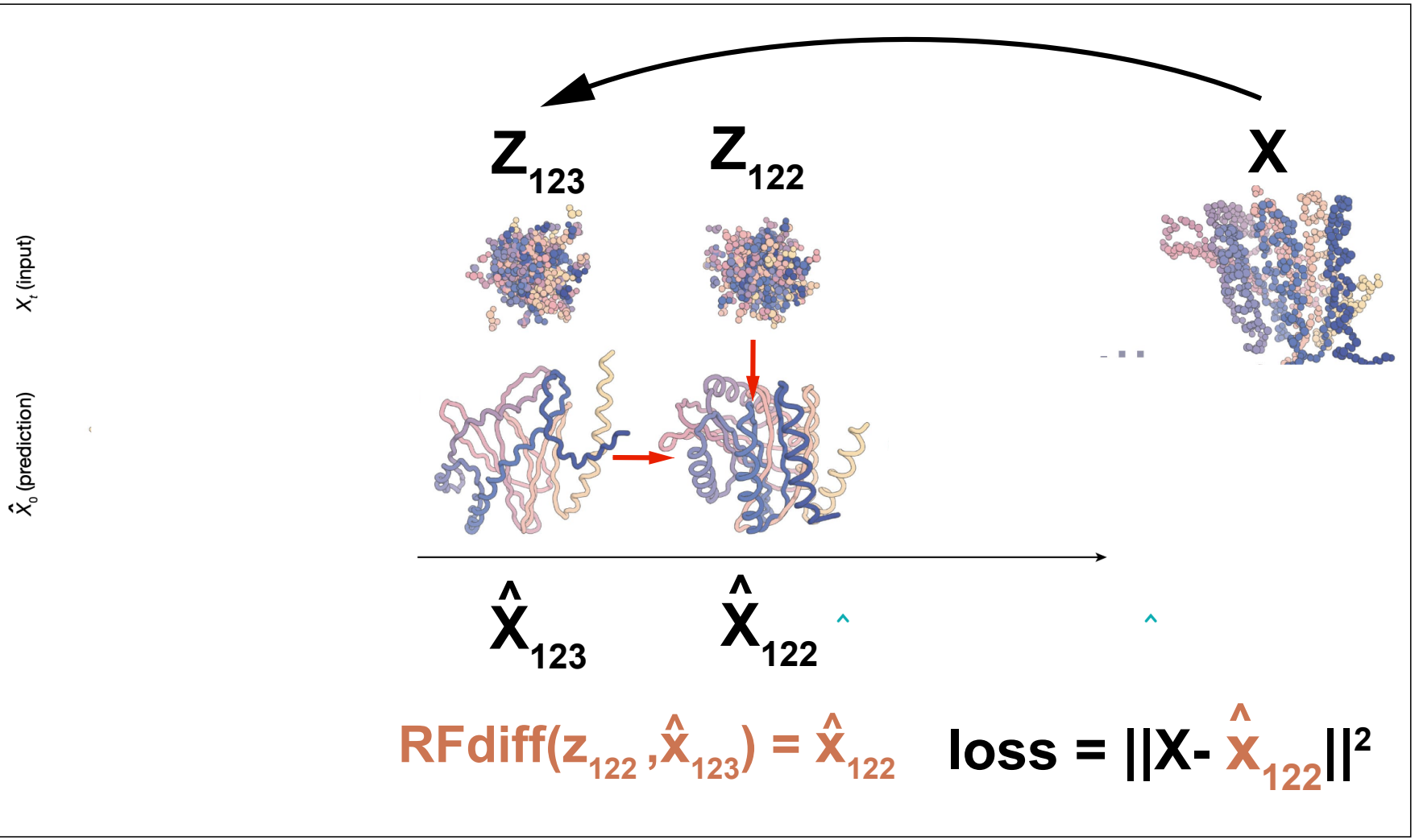

Z_{123}
X
X_t (input)
$\hat{X}_0$ (prediction)
$\hat{X}_{123}$
RFdiff(z_{123}) = $\hat{x}_{123}$
loss = ||X- $\hat{x}_{123}$||²


Z_{t+1} = ForwardNoise(X,t+1)
Z_{123}
X
X_t (input)
$\hat{X}_0$ (prediction)


Z_{123}
X
X_t (input)
$\hat{X}_0$ (prediction)
...
$\hat{X}_{123}$
$\hat{x}_{t+1}$ = RFDiffusion(Z_{t+1})
RFdiff(z_{123}) = $\hat{x}_{123}$


X_t (input)
$\hat{X}_0$ (prediction)
Z_{123}
Z_{122}
X
$\hat{X}_{123}$
$Z_t = \text{ReverseStep}(Z_{t-1}, \hat{x}_{t-1})$
$z_{122} = \text{interpol}(z_{123}, \hat{x}_{123})$


X_t (input)
$\hat{X}_0$ (prediction)
Z_{123}
Z_{122}
X
$\hat{X}_{123}$
$\hat{X}_{122}$
$\hat{x}_t = \text{RFDiffusion}(Z_t, \hat{x}_{t+1})$
$\text{RFdiff}(z_{122}, \hat{x}_{123}) = \hat{x}_{122}$


X_t (input)
$\hat{X}_0$ (prediction)
Z_{123}
Z_{122}
X
$\hat{X}_{123}$
$\hat{X}_{122}$
$RFdiff(z_{122}, \hat{x}_{123}) = \hat{x}_{122}$
$loss = ||X - \hat{x}_{122}||^2$

Benchmarking

Flow Matching

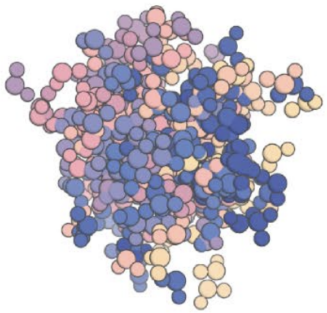
vs Diffussion

What does Flow Matching do differently?

Diffusion

noise → sample

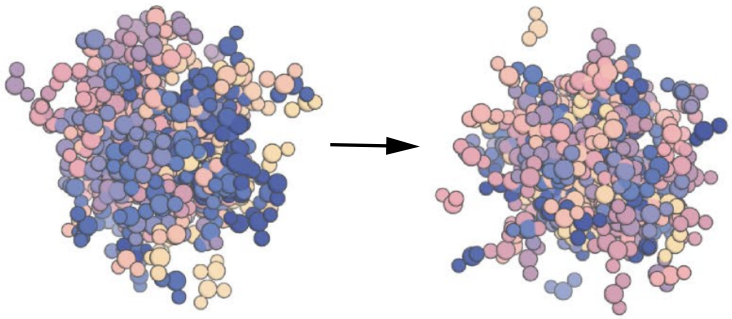
generative process



Diffusion

noise → sample

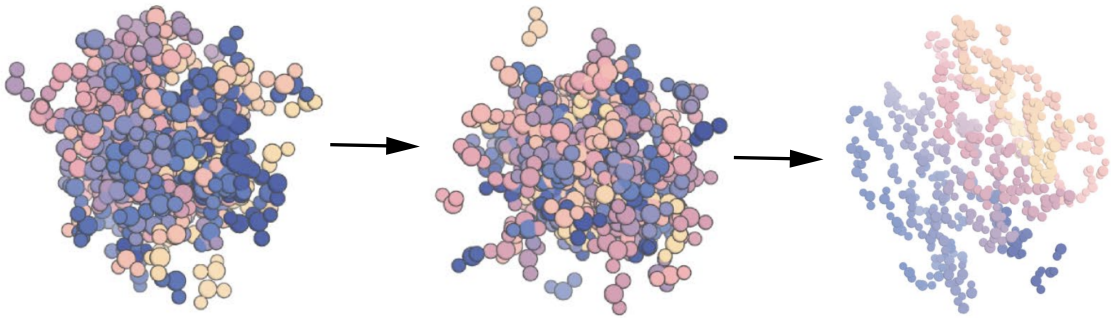
generative process



Diffusion

noise → sample

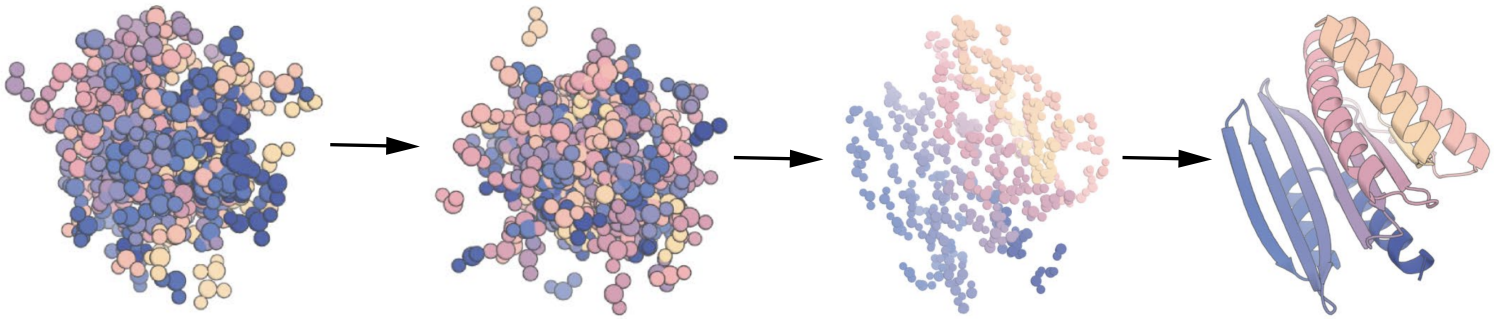
generative process



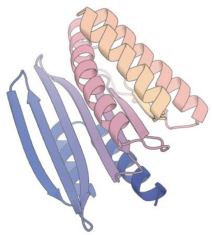
Diffusion

noise → sample

generative process



DIFFUSION Training



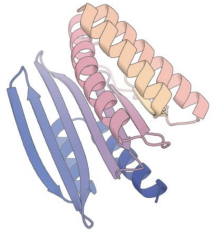
data

t=1

DIFFUSION Training

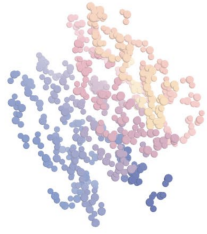
$t \sim \text{Uniform}[1...0]$

forward diffusion

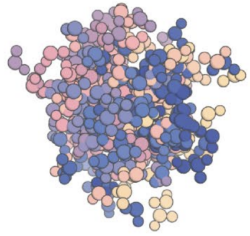


data x_1

$t=1$



x_t



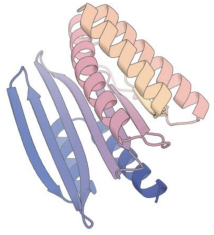
noise x_0

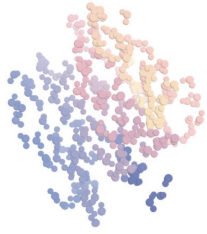
$t=0$

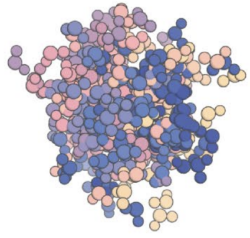
DIFFUSION Training

$t \sim \text{Uniform}[1...0]$

forward diffusion







data x_1
$t=1$

$$x_t = \sqrt{\alpha_t}\, x_1 + \sqrt{1-\alpha_t}\, \epsilon$$

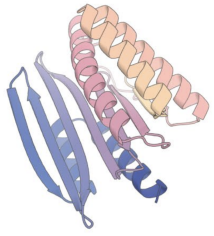
$\epsilon \sim N(0,I)$

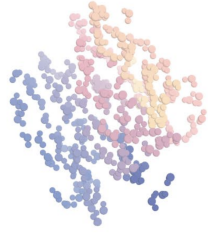
noise x_0
$t=0$

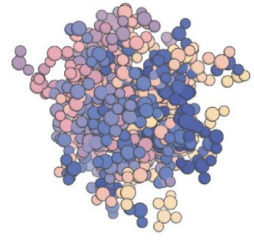
DIFFUSION Training

t ~ Uniform[1...0]

forward diffusion







data X_1
t=1

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon$$

N(0,I)

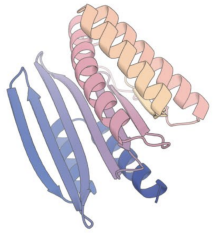
noise X_0
t=0

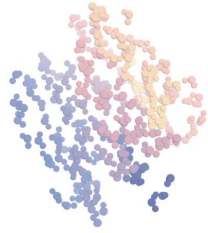
$$\text{var}(X_t) = \alpha_t \,\text{var}(X_1) + (1-\alpha_t)\,\text{var}(N(0,1))$$

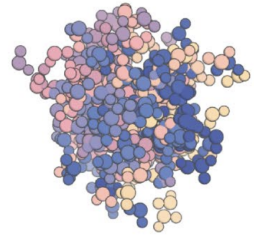
DIFFUSION Training

t ~ Uniform[1...0]

forward diffusion







N(0,I)

data X_1
t=1

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon$$

noise X_0
t=0

$$\text{var}(X_t) = \alpha_t \,\text{var}(X_1) + (1-\alpha_t)\,\text{var}(N(0,1))$$

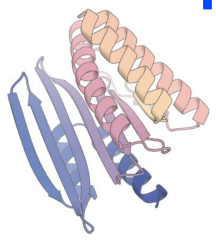
If $\text{var}(X_1) = 1$ then $\text{var}(X_t) = \text{var}(X_1) = 1$

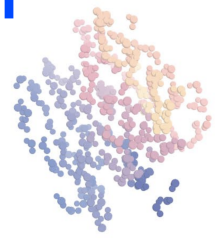
DIFFUSION Training

$t \sim \text{Uniform}[1 \ldots 0]$

forward diffusion

backward diffusion





data X_1

$t=1$

X_t

$N(0,I)$

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon$$

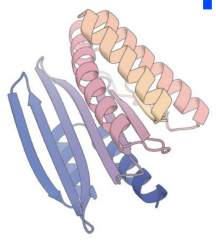
DIFFUSION Training

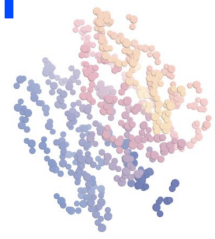
$t \sim \text{Uniform}[1...0]$

Neural Network

forward diffusion

backward diffusion $\epsilon_\Theta(X_t)$





data X_1

$t=1$

X_t

$N(0,I)$

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon$$

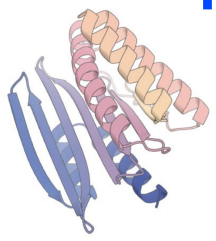
DIFFUSION Training

$t \sim \text{Uniform}[1...0]$

Neural Network

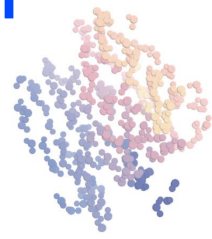
forward diffusion

backward diffusion $\epsilon_\Theta(X_t)$



data X_1

$t=1$



X_t

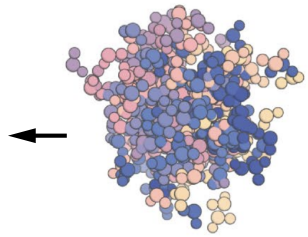
$N(0,I)$

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon$$

$$\text{Loss} = \| \epsilon - \epsilon_\Theta(X_t) \|^2$$

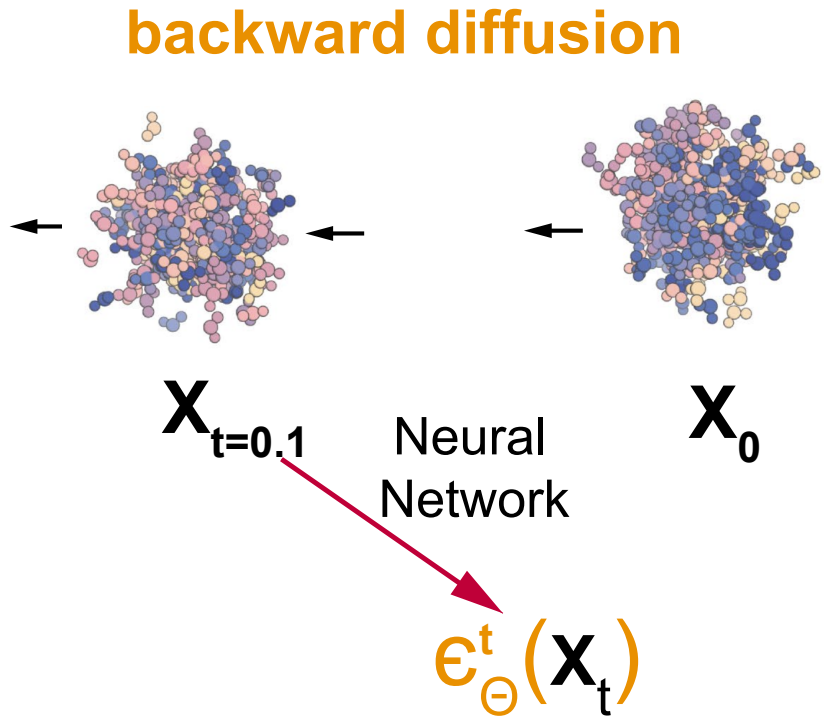
DIFFUSION Sampling

backward diffusion

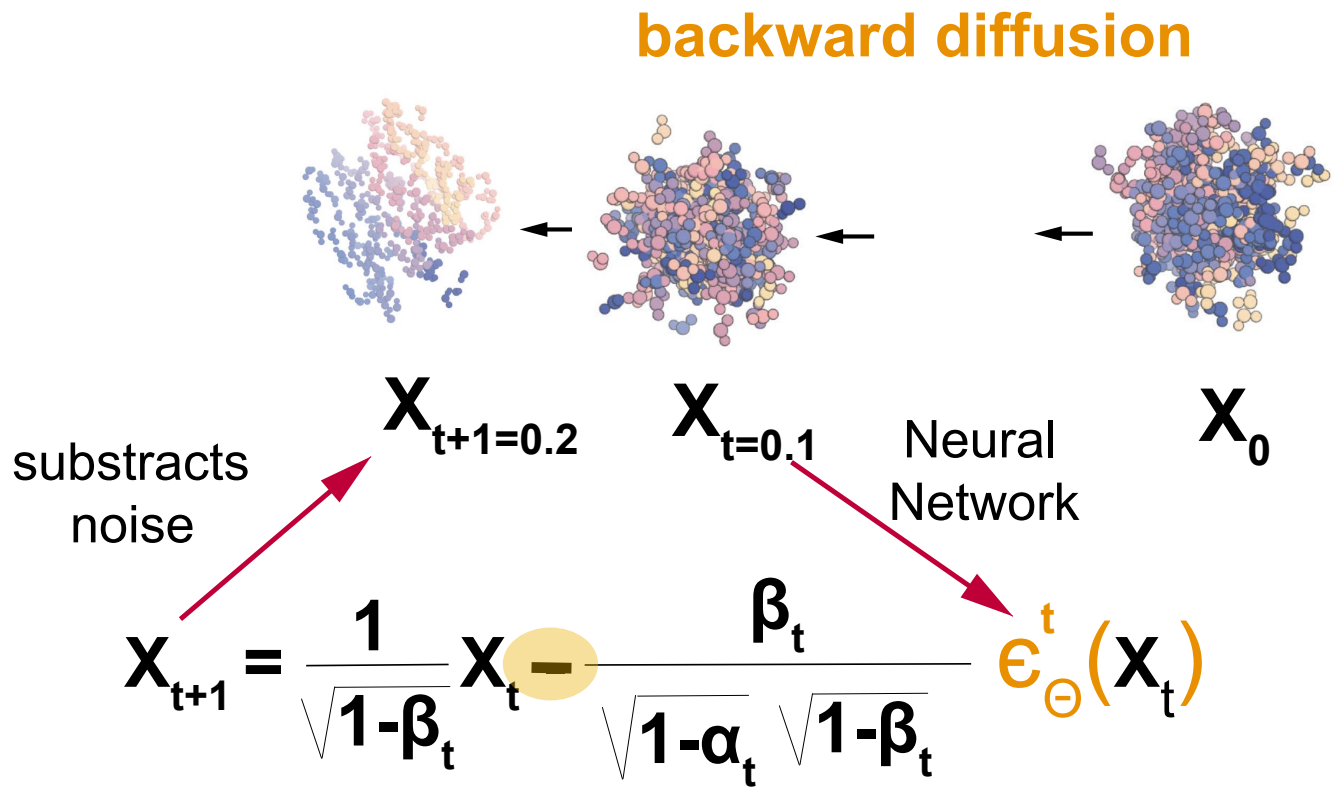


X_0

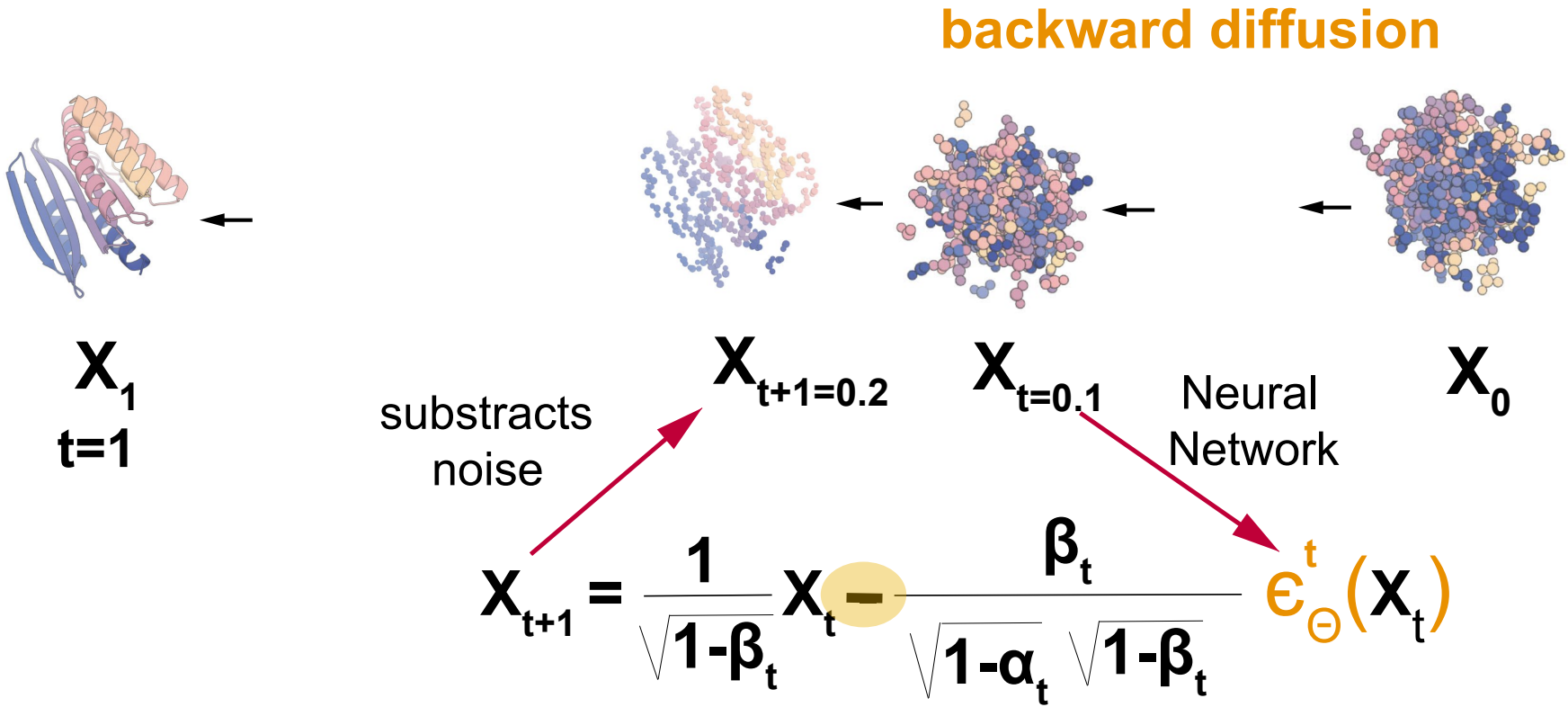
DIFFUSION Sampling



DIFFUSION Sampling



DIFFUSION Sampling



Diffusion Summary

TRAINING

For all real images

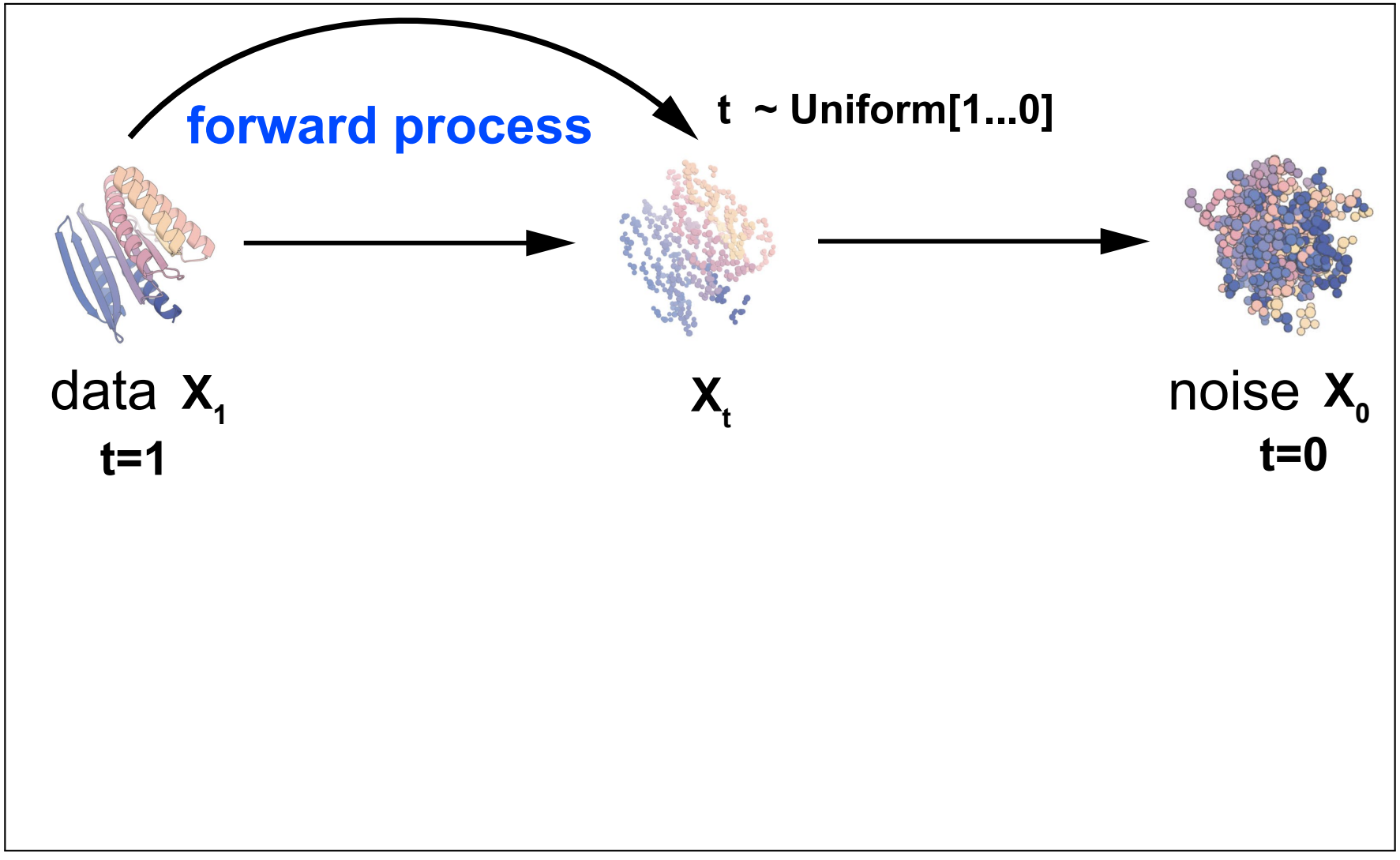
1. get real image X_1
2. sample time $t \in [0, 1]$
3. Forward diffusion
   Create noise $X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1 - \alpha_t}\, \epsilon^t$
4. Backward diffusion Predict noise with NN $\epsilon_\theta^t(X_t)$
5. Loss MSE $= ||\epsilon_\theta^t(X_t) - \epsilon^t||^2$

SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   2.1 Backward diffusion Predict noise with NN $\epsilon_\theta^t(X_k)$ for $t = k/T$
   2.2 $X_{k+1}^{pred} = \frac{1}{\sqrt{1-\beta_t}}\, X_k^{pred} - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\, \epsilon_\theta^t(X_k^{pred})$
   $X_1 = X_{k=T}^{pred}$

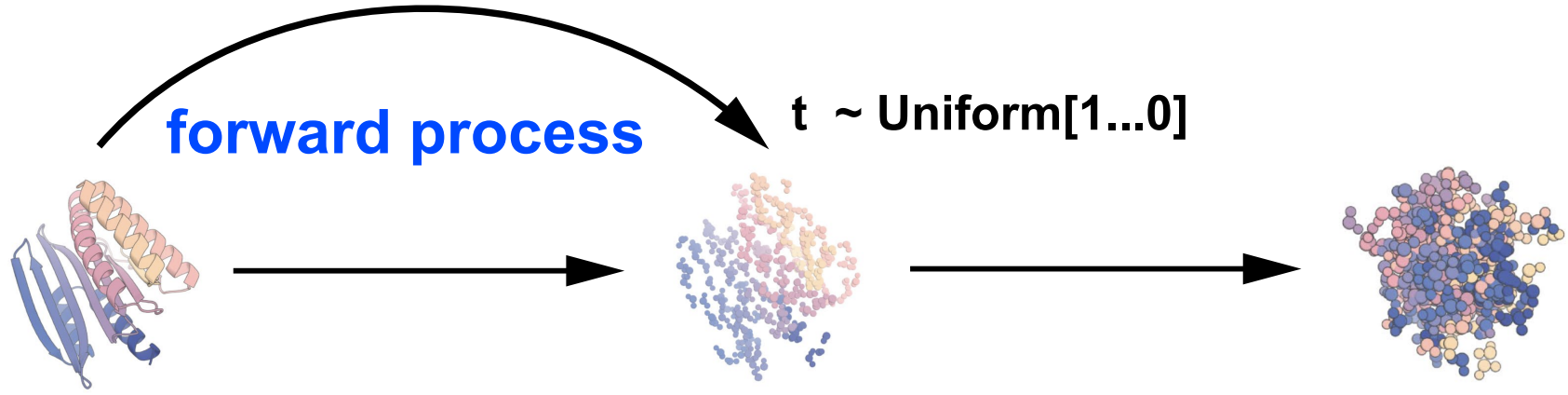
Flow Matching

Path from X_1 to X_0 is deterministic



Flow Matching

Path from X_1 to X_0 is deterministic



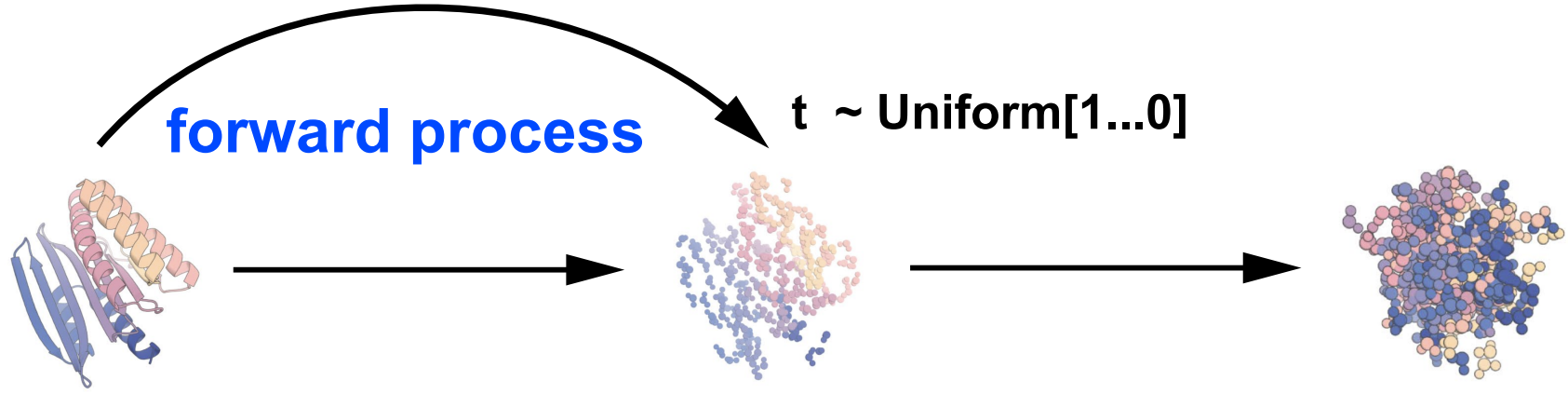
N(0,I)

DIFFUSION

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon_t$$ stochastic

Flow Matching

Path from X_1 to X_0 is deterministic



N(0,I)

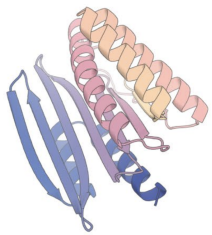
DIFFUSION

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon_t$$ **stochastic**

FLOW MATCHING **deterministic**

$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$

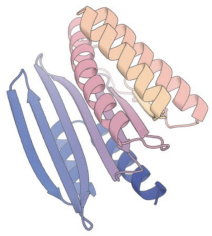
FLOW MATCHING Training



data

t=1

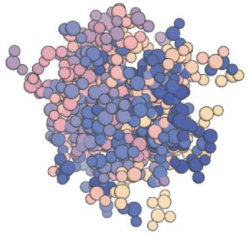
FLOW MATCHING Training



data X_1

t=1

Sample

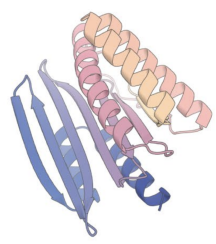


$X_0 \sim N(0,1)$

t=0

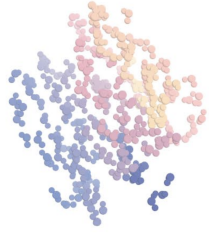
FLOW MATCHING Training

$t \sim \text{Uniform}[1...0]$



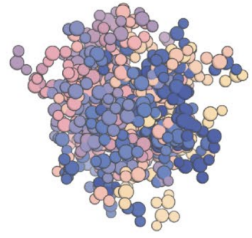
data X_1

$t=1$



X_t

Sample

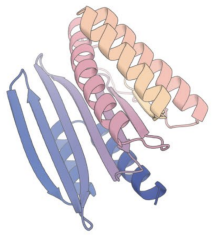


$X_0 \sim N(0,1)$

$t=0$

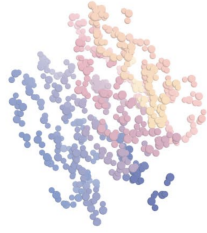
FLOW MATCHING Training

$t \sim \text{Uniform}[1...0]$



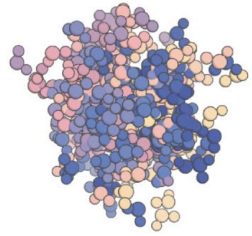
data X_1

$t=1$



X_t

Sample

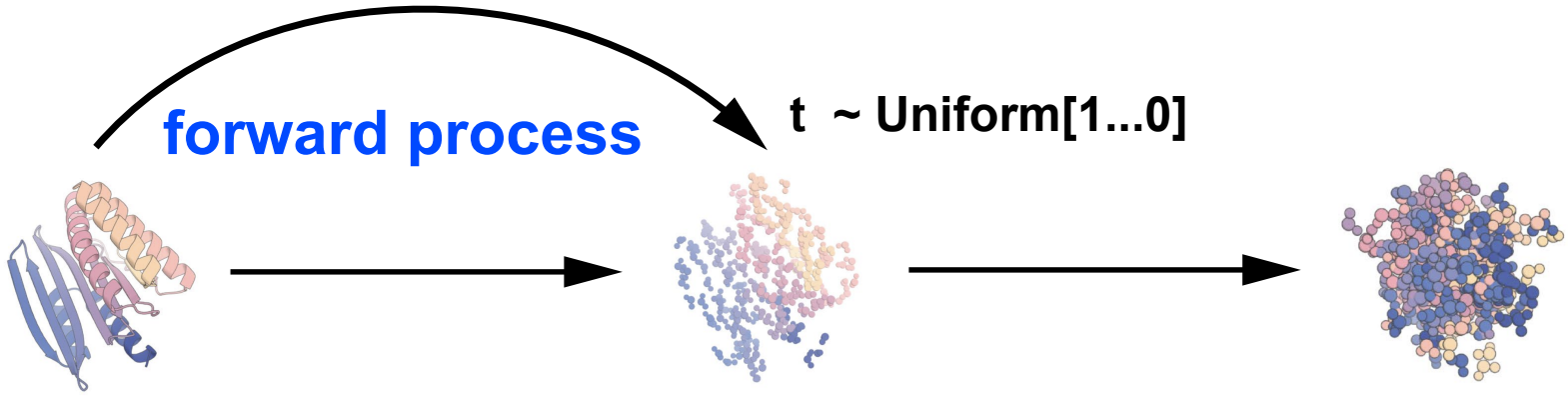


$X_0 \sim N(0,1)$

$t=0$

$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$

FM = much simpler diffusion process



data x_1
t=1

x_t

noise x_0
t=0

N(0,I)

DIFFUSION

$$X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon_t$$ stochastic on t

FLOW MATCHING deterministic on t>0

$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$ stochastic only on X_0

N(0,I)

From your Physics 101

- A constant velocity v trajectory

$$x = x_0 + v\,t$$

  is equivalent to

$$\frac{dx}{dt} = v$$

- Generalizes to

$$\frac{dx}{dt} = v(t)$$

  For example: $v(t) = \sqrt{t}$

- Further generalizes to a **flow** $v(x,t)$

$$\frac{dx}{dt} = v(x,t)$$

  For example: $v(x,t) = x\,\sqrt{t}$

Ordinary Differential Equation (OED)

$$\frac{dx}{dt} = v(x,t)$$

Has the solution

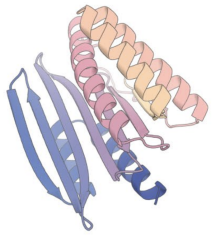
$$x_1 = x_0 + \int_0^1 v(x,t)dt$$

Which can be solved numerically (Euler Method)
$k = 1, ..., T$ and $t = k/T$

$$x_k = x_{k-1} + \frac{1}{T}v(x, k-1)$$

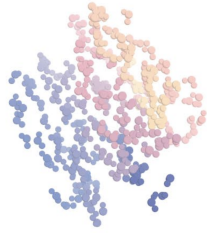
FLOW MATCHING Training

$t \sim \text{Uniform}[1 \ldots 0]$

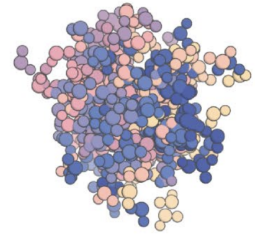


data X_1

$t=1$



X_t



$X_0 \sim N(0,1)$

$t=0$

$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$

$$\frac{dX_t}{dt} = V(X_t) = X_1 - X_0$$

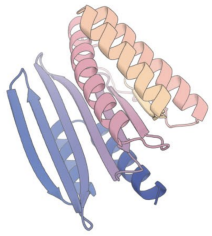
constant flow for all t's

FLOW MATCHING Training

$t \sim \text{Uniform}[1...0]$

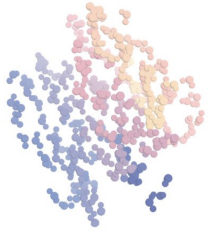
backward NN

$V_\Theta(X_t)$

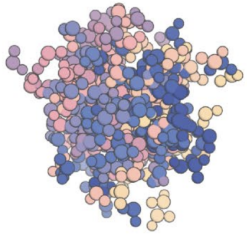


data X_1

$t=1$



X_t



$X_0 \sim N(0,1)$

$t=0$

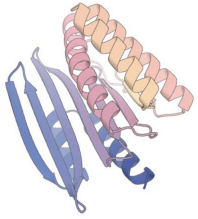
$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$

$$\frac{dX_t}{dt} = V(X_t) = X_1 - X_0$$

FLOW MATCHING Training

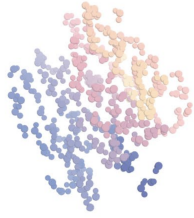
$t \sim \text{Uniform}[1 \ldots 0]$

backward NN



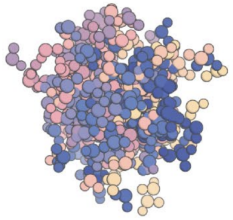
data X_1

$t=1$



X_t

$V_\Theta(X_t)$



$X_0 \sim N(0,1)$

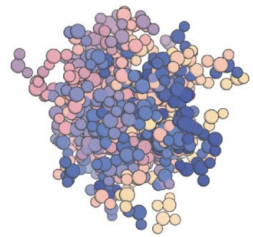
$t=0$

$$X_t = t \cdot X_1 + (1-t) \cdot X_0$$

$$\frac{dX_t}{dt} = V(X_t) = X_1 - X_0$$

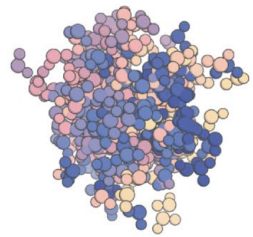
$$\text{Loss} = \| V(X_t) - V_\Theta(X_t) \|^2$$

FLOW MATCHING Sampling



$X_0 \sim N(0,1)$

FLOW MATCHING Sampling

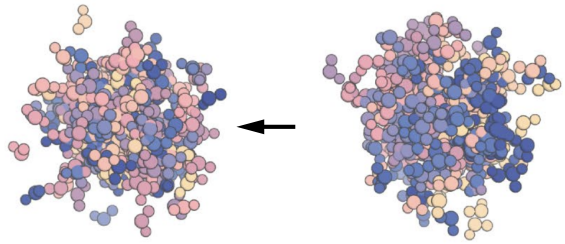


X_0

Neural Network

$V_\Theta(X_0)$

FLOW MATCHING Sampling

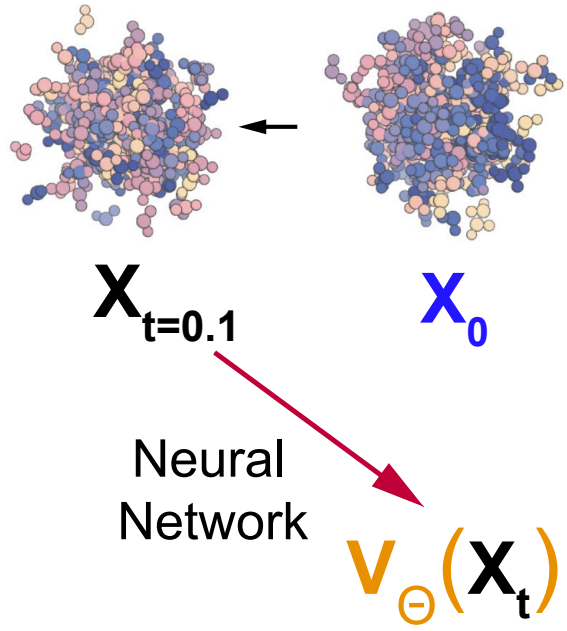


$X_{t=0.1}$ X_0

T=10

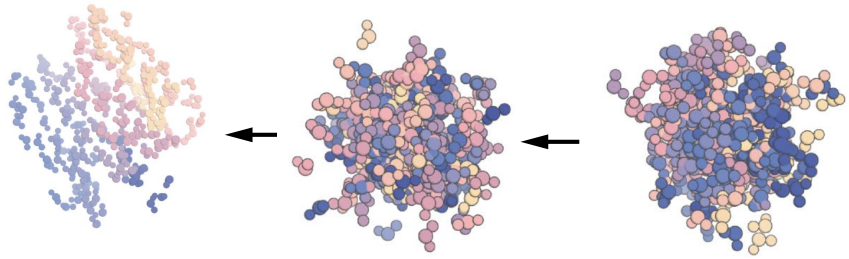
$$X_{0.1} = X_0 + 0.1\, V_{\ominus}(X_0)$$

FLOW MATCHING Sampling



T=10

FLOW MATCHING Sampling



$X_{t+1=0.2}$ $X_{t=0.1}$ X_0

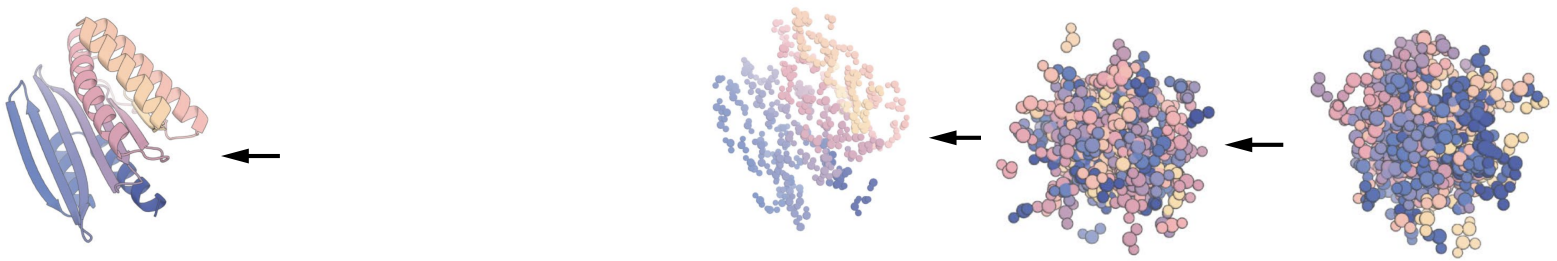
T=10

$$X_{t+1} = X_t + 0.1\, V_\Theta(X_t)$$

Neural Network

$V_\Theta(X_t)$

FLOW MATCHING Sampling



X_1
t=1

$X_{t=0.2}$ $X_{t=0.1}$ X_0
t=0

$$X_1 = X_{0.9} + 0.1\, V_\Theta(X_{0.9})$$

T=10

Flow Matching Summary

TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0, 1)$
3. sample time $t \in [0, 1]$
4. Forward interpolate
   $X_t = t\, X_1 + (1 - 1)\, X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   2.2 Using Euler method:
   $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T}\, V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
   $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$

Diffusion Summary

TRAINING

For all real images

1. get real image X_1
2. sample time $t \in [0, 1]$
3. Forward diffusion
   Create noise $X_t = \sqrt{\alpha_t}\, X_1 + \sqrt{1-\alpha_t}\, \epsilon^t$
4. Backward diffusion Predict noise with NN $\epsilon_\theta^t(X_t)$
5. Loss MSE $= ||\epsilon_\theta^t(X_t) - \epsilon^t||^2$

SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   2.1 Backward diffusion Predict noise with NN $\epsilon_\theta^t(X_k)$ for $t = k/T$
   2.2 $X_{k+1}^{pred} = \frac{1}{\sqrt{1-\beta_t}}\, X_k^{pred} - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\, \epsilon_\theta^t(X_k^{pred})$
   $X_1 = X_{k=T}^{pred}$

Flow Matching Summary

TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0, 1)$
3. sample time $t \in [0, 1]$
4. Forward interpolate
   $X_t = t\, X_1 + (1-1)\, X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   2.2 Using Euler method:
   $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T}\, V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
   $X_1 = X_0 + \int_0^1 V_\theta(X_t) dt = X_{k=T}^{pred}$

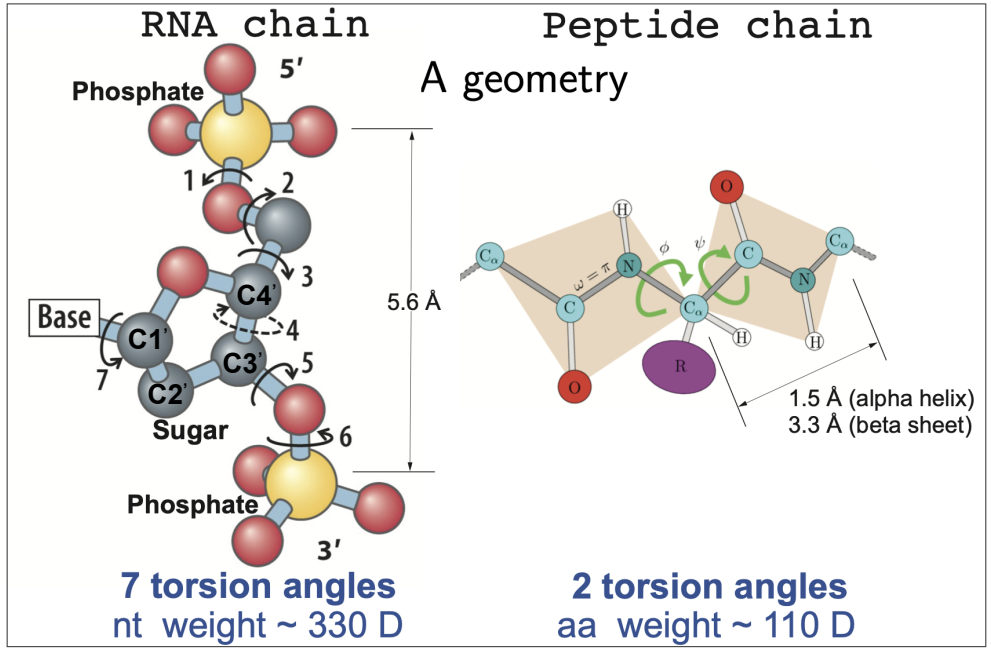
RiboGen: RNA Sequence and Structure Co-Generation with Equivariant MultiFlow

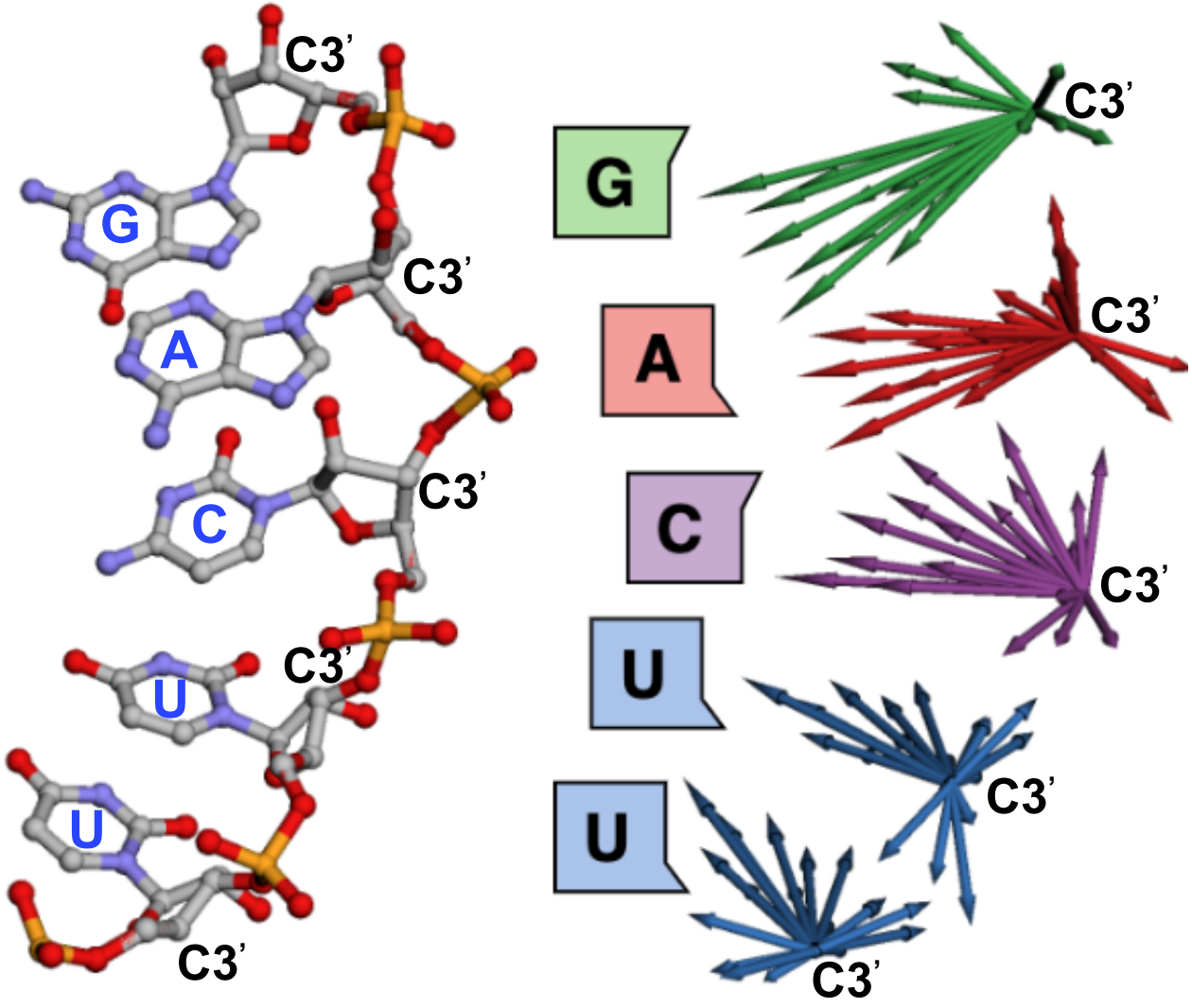
Dana Rubin
MIT CSAIL
MIT Media Lab, Molecular Machines
danaru@mit.edu

Allan dos Santos Costa
Center for Bits and Atoms
MIT Media Lab, Molecular Machines
allanc@mit.edu

Manvitha Ponnapati
Center for Bits and Atoms
MIT Media Lab, Molecular Machines

Joseph Jacobson
Center for Bits and Atoms
MIT Media Lab, Molecular Machines

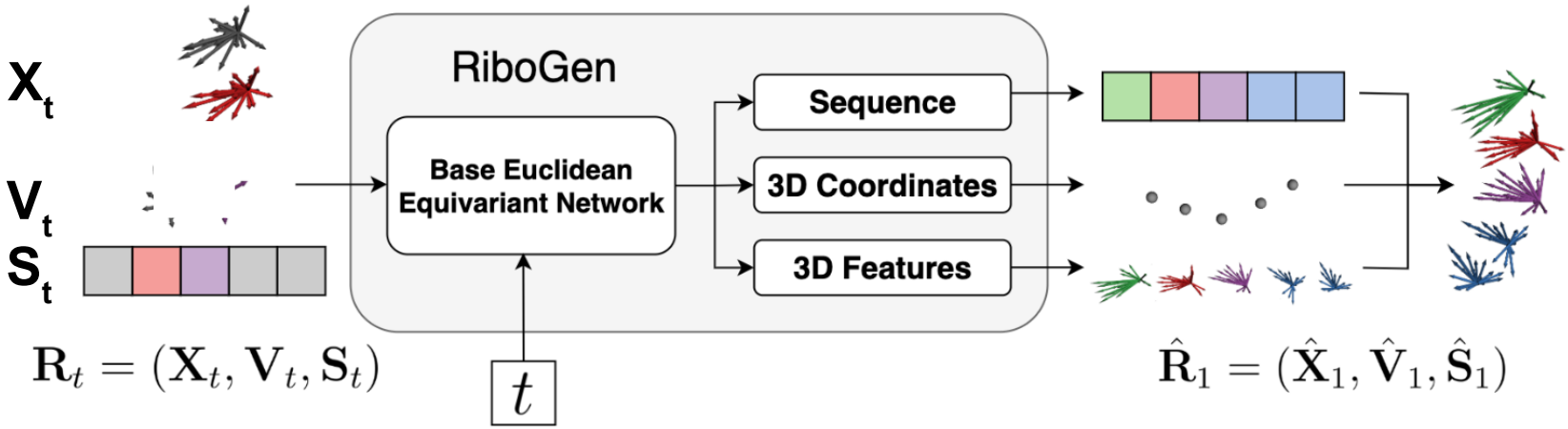


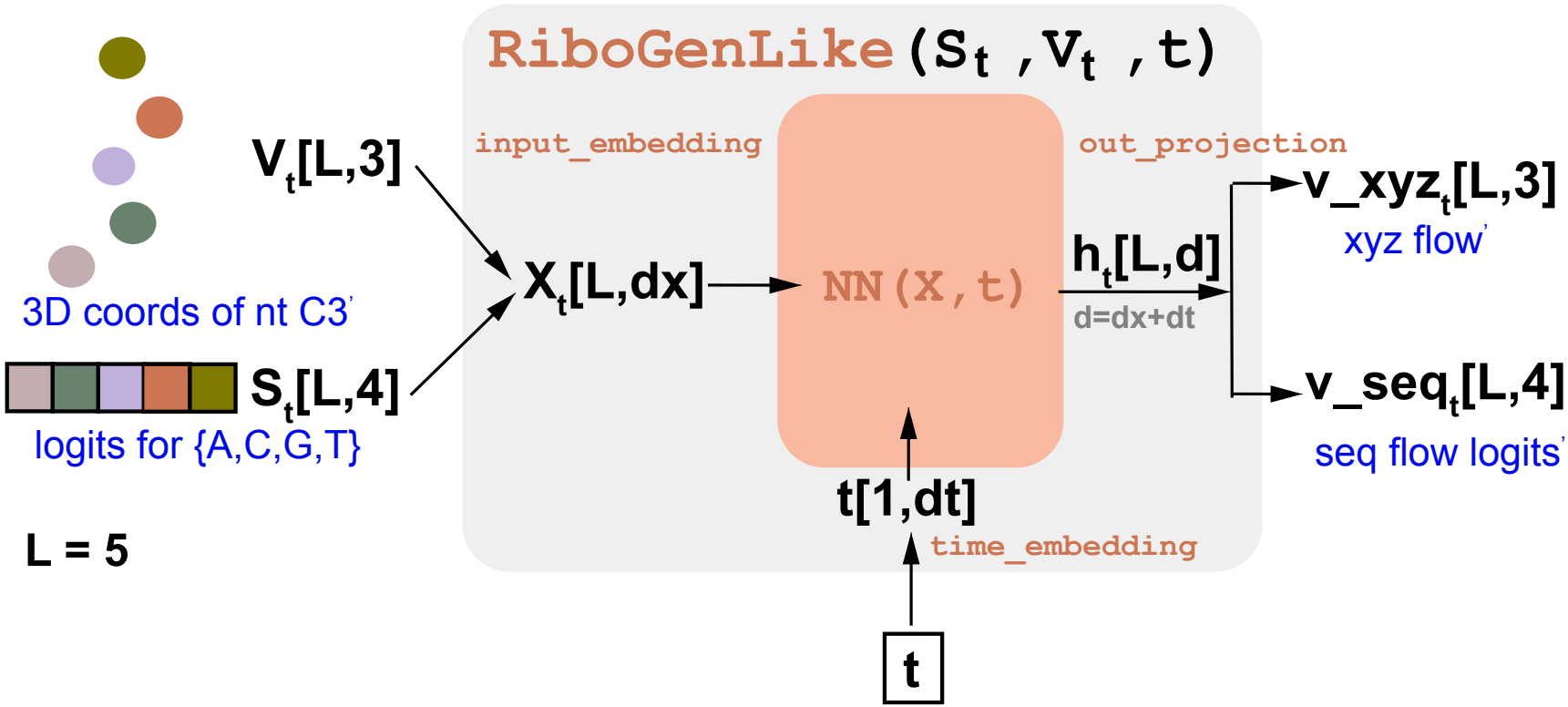


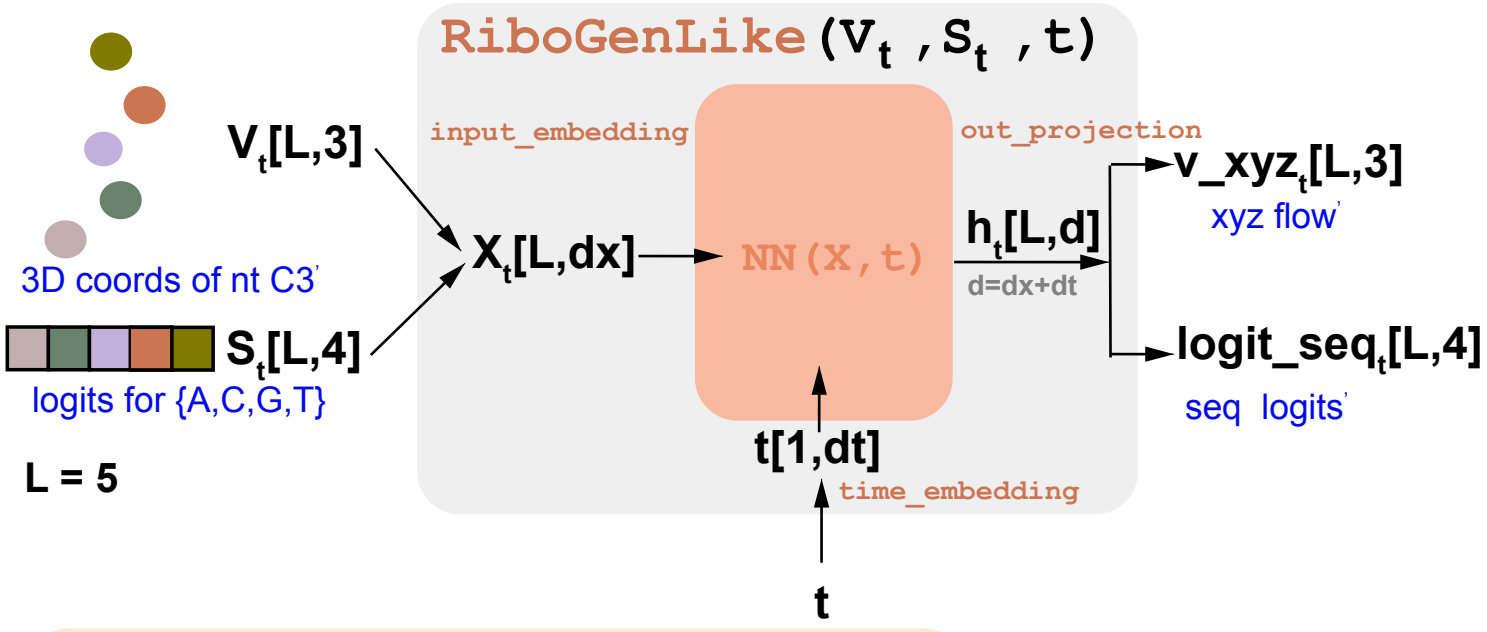
X[L,3]
global coords
(x,y,z) of C3'

S[L]
{A,C,G,T}

V[L,24,3]
local coords
all nt atoms


X_t
V_t
S_t
RiboGen
Base Euclidean Equivariant Network
Sequence
3D Coordinates
3D Features
$\mathbf{R}_t = (\mathbf{X}_t, \mathbf{V}_t, \mathbf{S}_t)$
t
$\hat{\mathbf{R}}_1 = (\hat{\mathbf{X}}_1, \hat{\mathbf{V}}_1, \hat{\mathbf{S}}_1)$

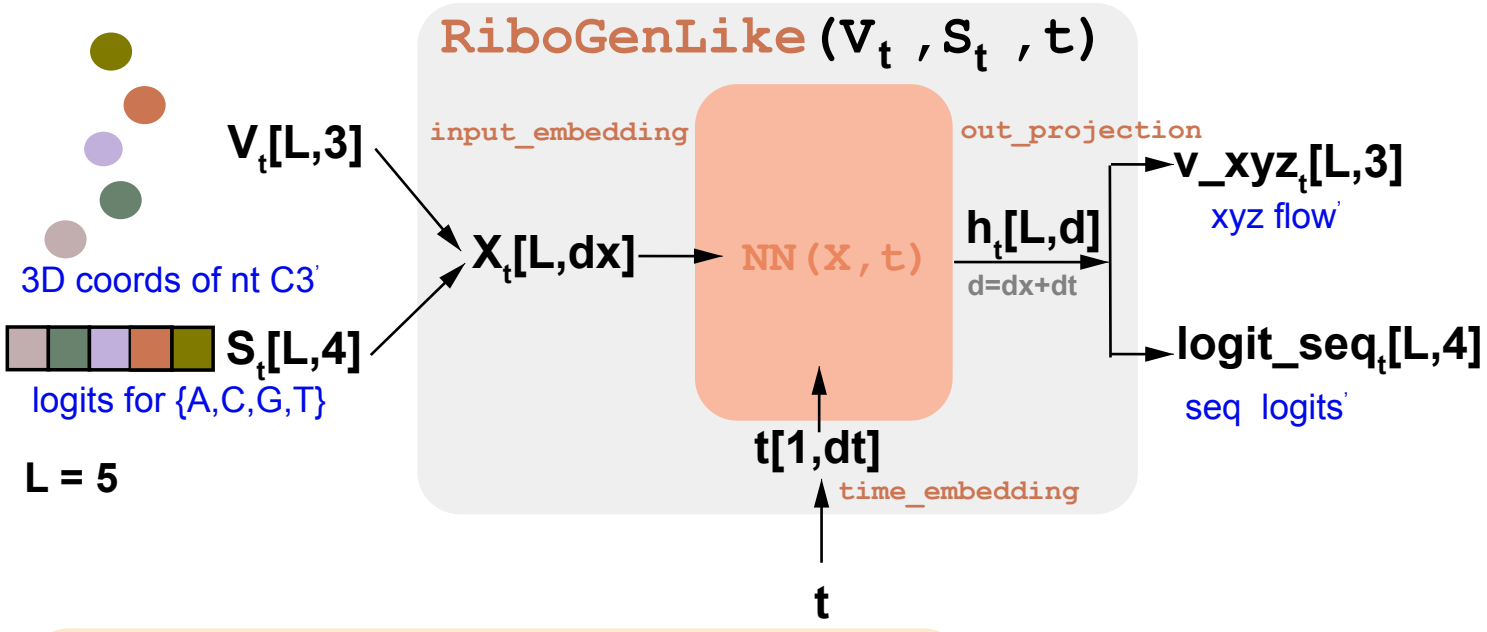

RiboGenLike(S_t , V_t , t)
input_embedding
out_projection
V_t[L,3]
3D coords of nt C3'
X_t[L,dx]
NN(X,t)
h_t[L,d]
d=dx+dt
v_xyz_t[L,3]
xyz flow'
S_t[L,4]
logits for {A,C,G,T}
v_seq_t[L,4]
seq flow logits'
L = 5
t[1,dt]
time_embedding
t



TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

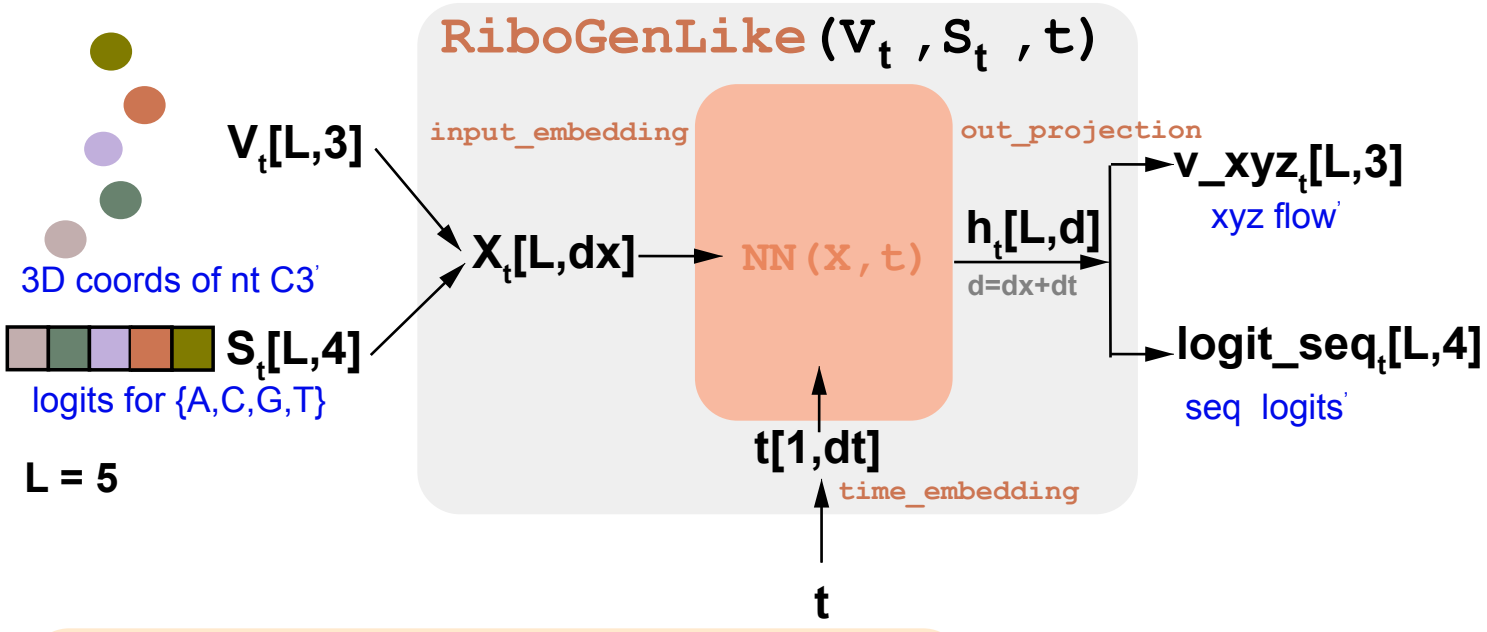


TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])



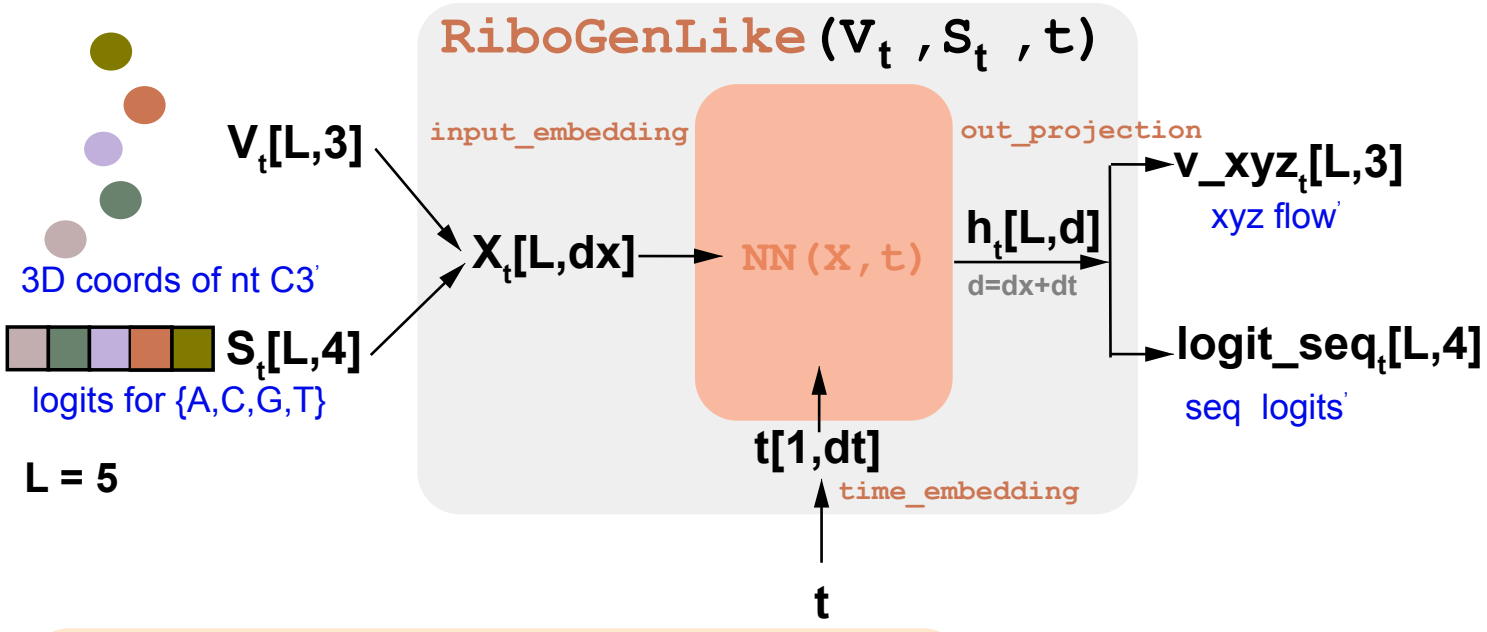
TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical



TRAINING

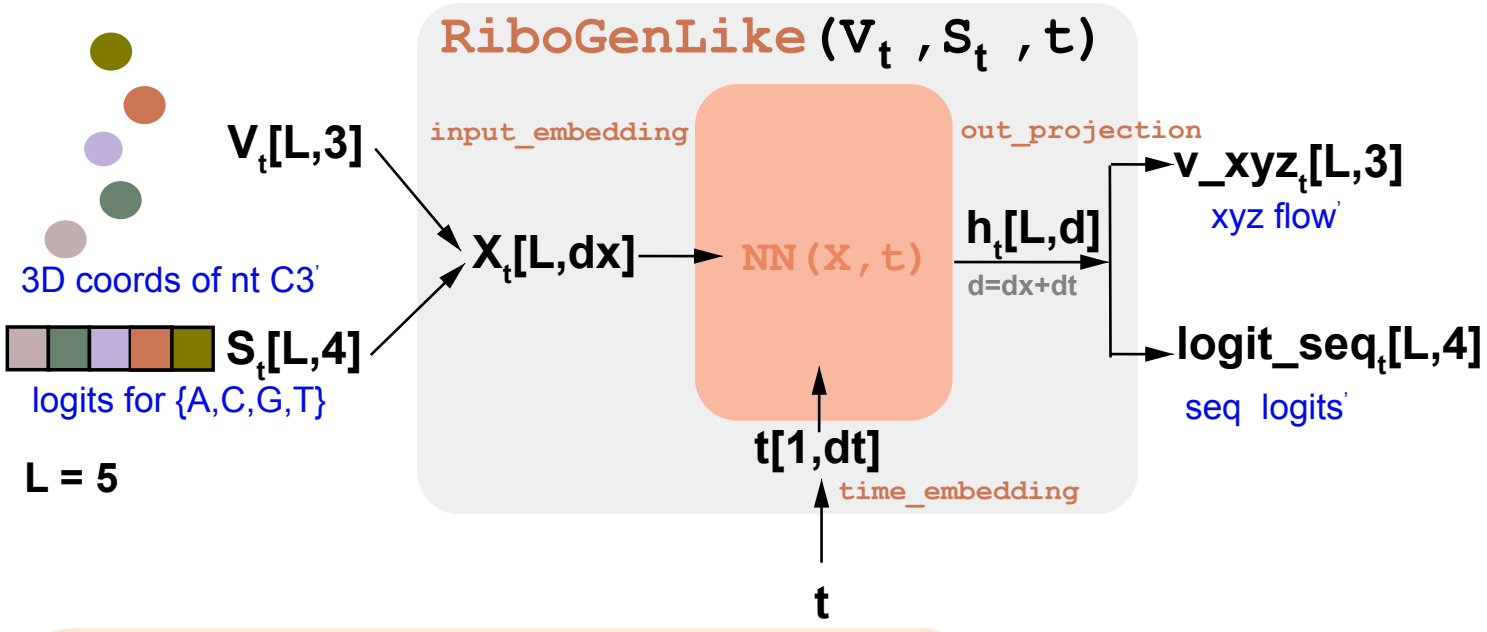
For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

t



TRAINING

For all real images

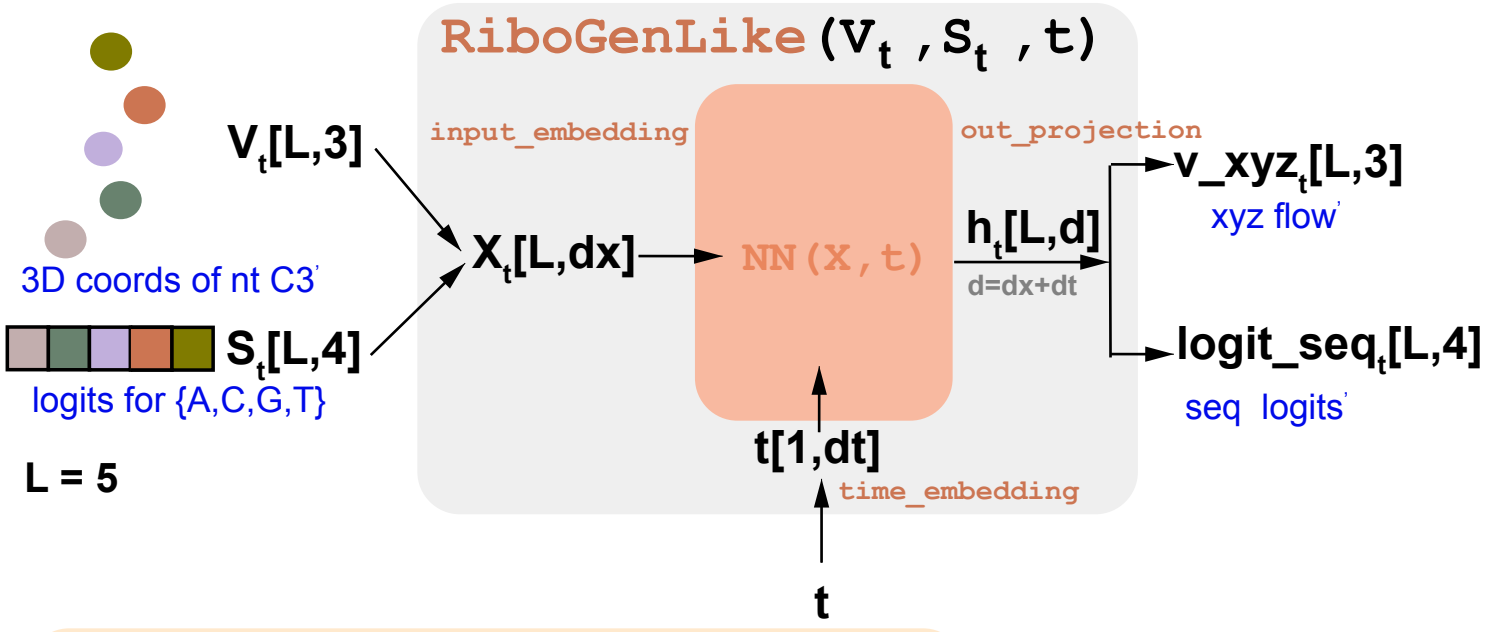
1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

t

V_t[L,3] = tV_1[L,3] + (1-t)V_0[L,3]



TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

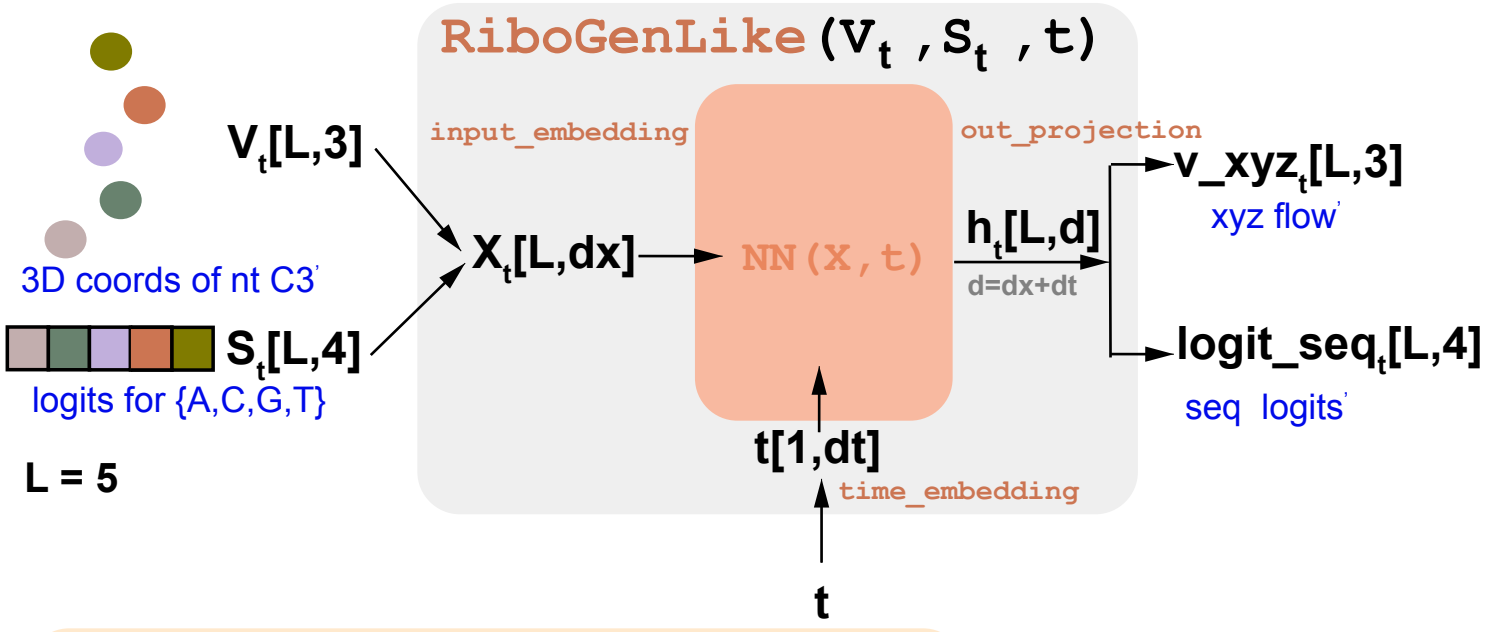
X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

t

V_t[L,3] = tV_1[L,3] + (1-t)V_0[L,3]

S_t[L] = bern(t)S_1[L] + (1-bern(t))S_0[L]



TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$ = **RiboGenLike (V_t , S_t , t)**
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

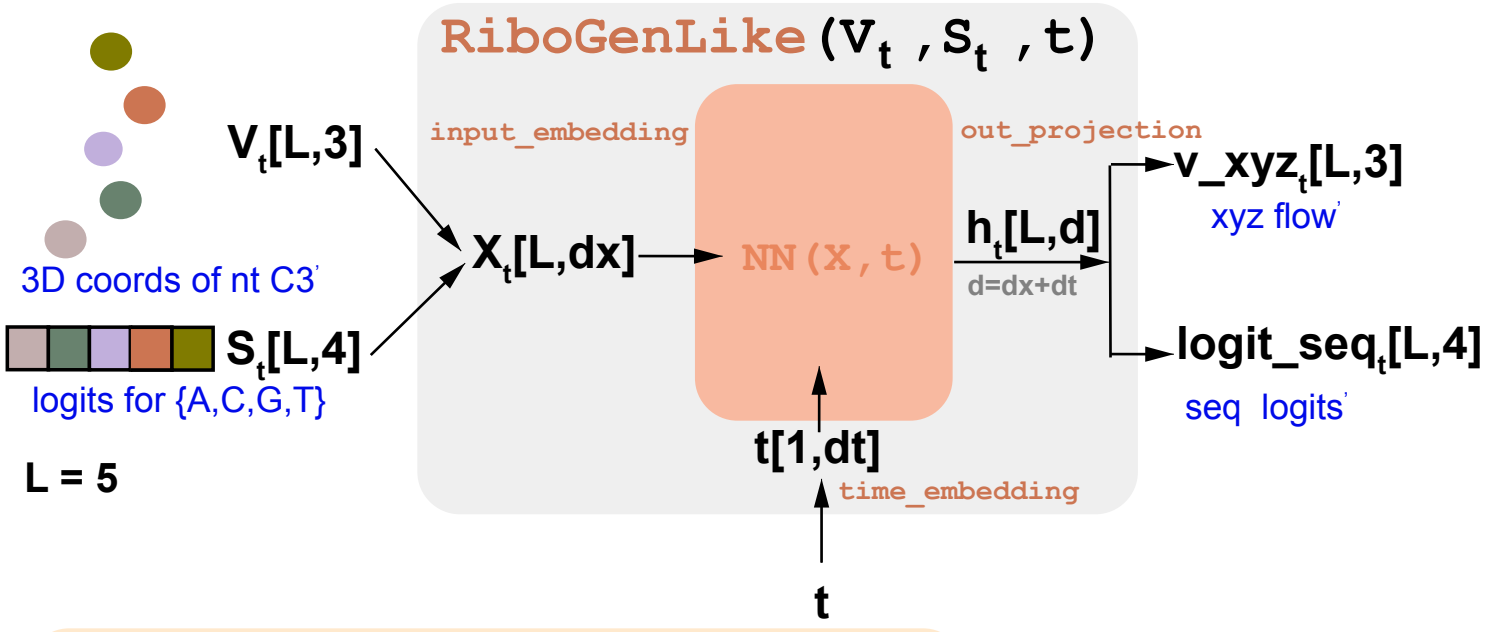
X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

t

V_t[L,3] = tV_1[L,3] + (1-t)V_0[L,3]

S_t[L] = bern(t)S_1[L] + (1-bern(t))S_0[L]



TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$ = **RiboGenLike (V_t , S_t , t)**
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

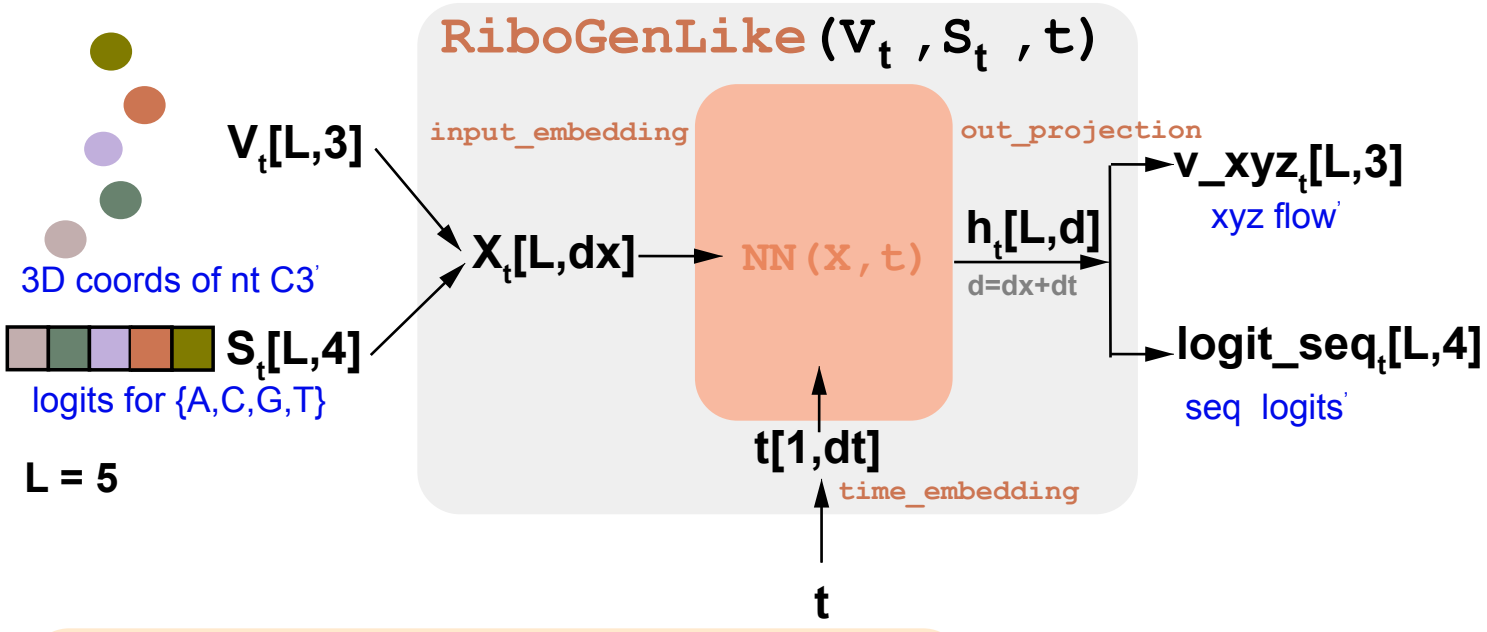
t

V_t[L,3] = tV_1[L,3] + (1-t)V_0[L,3]

S_t[L] = bern(t)S_1[L] + (1-bern(t))S_0[L]

2 Losses'

MSE(v_xyz$_t$[L,3], V_1[L,3]-V_0[L,3])



TRAINING

For all real images

1. get real image X_1
2. sample image $X_0 \sim N(0,1)$
3. sample time $t \in [0,1]$
4. Forward interpolate
   $X_t = t\,X_1 + (1-1)\,X_0$
   $V_t = X_1 - X_0$
5. Backward Predict flow with NN $V_\theta(X_t)$ = **RiboGenLike (V_t , S_t , t)**
6. Loss MSE $= ||V_\theta(X_t) - V_t||^2$

X_1 = cat(V_1[L,3], S_1[L,4])

V_0[L,3]~N(0,1) S_0[L] ~Uniform categorical

t

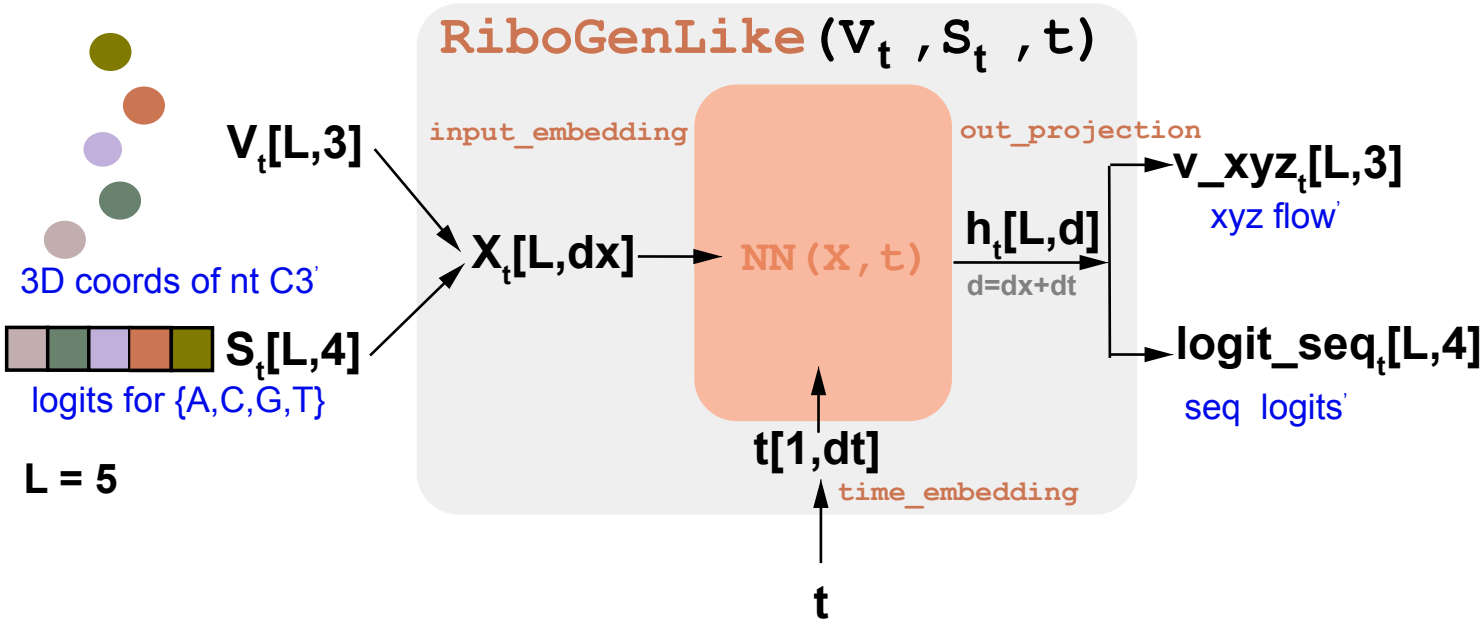
V_t[L,3] = tV_1[L,3] + (1-t)V_0[L,3]

S_t[L] = bern(t)S_1[L] + (1-bern(t))S_0[L]

2 Losses'

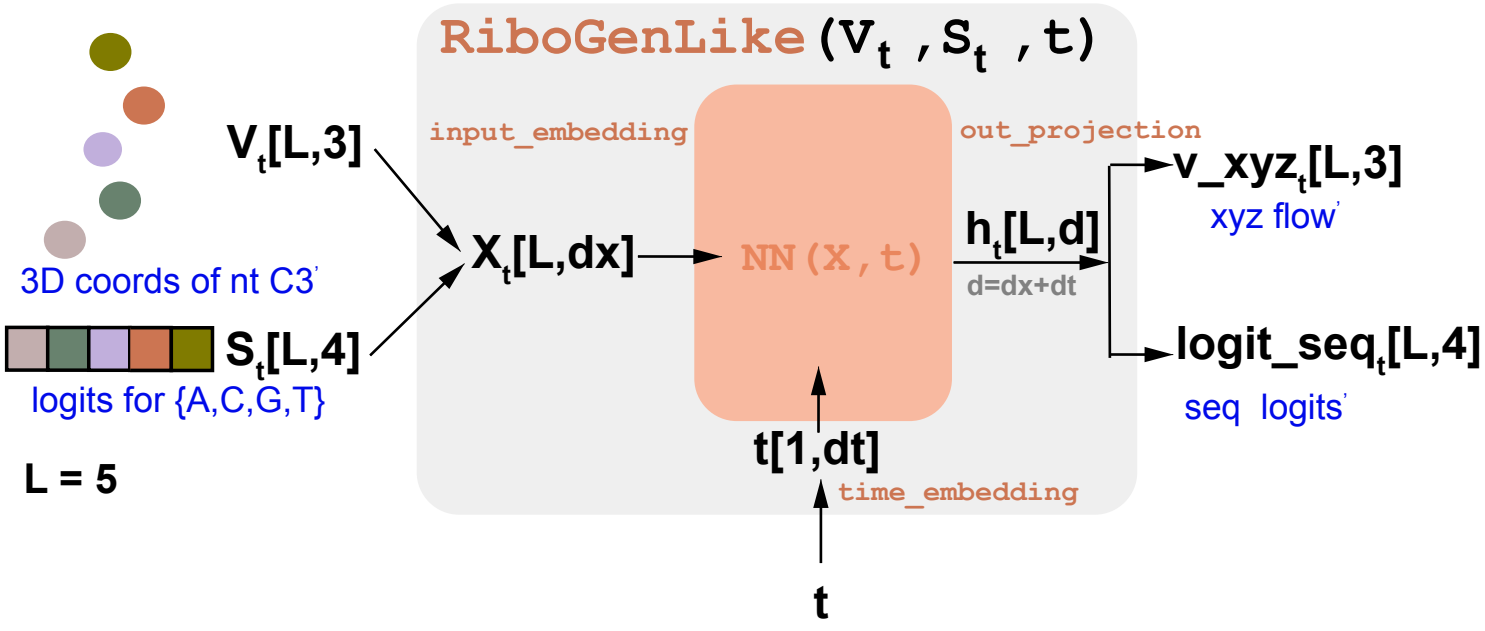
MSE(v_xyz_t[L,3], V_1[L,3]-V_0[L,3])

CrossEntropy($logit\_seq_t$[L,4], S_1[L,4])



SAMPLING

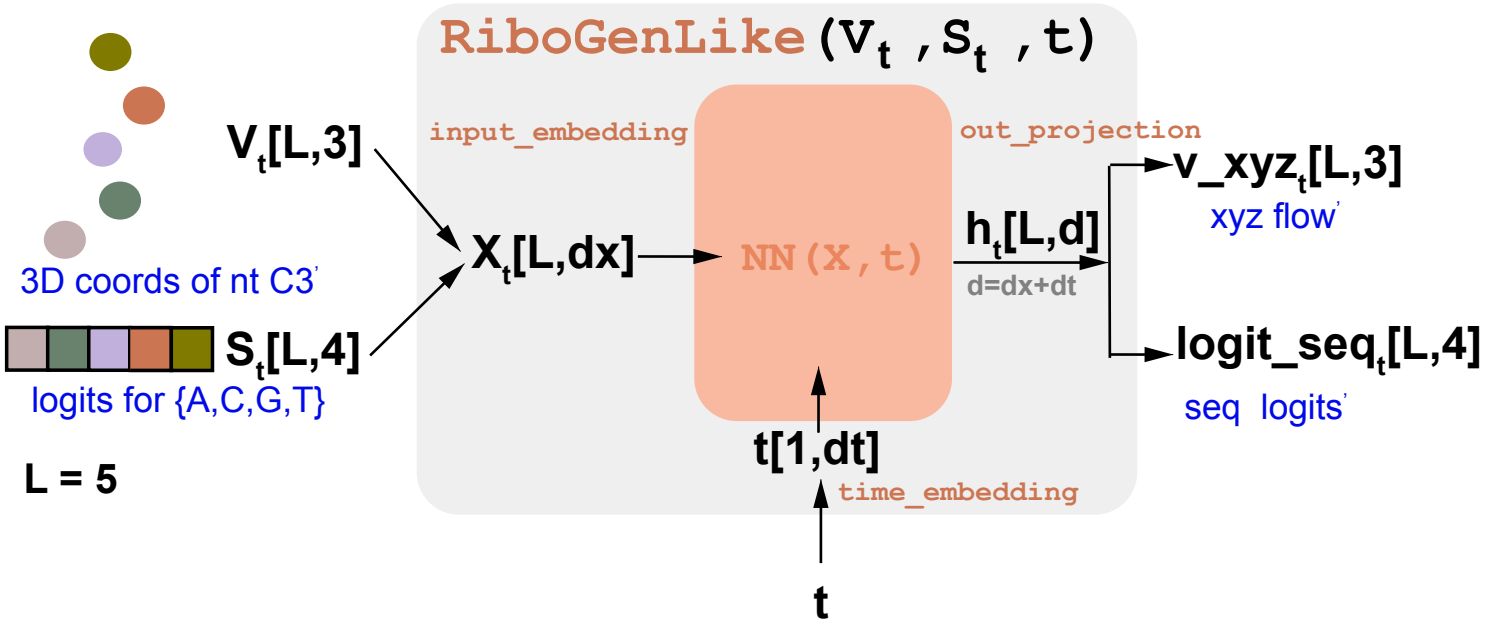
1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   - 2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   - 2.2 Using Euler method:
     $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T} V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
     $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$



SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   - 2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   - 2.2 Using Euler method:
     $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T} V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
     $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$

V_0[L,3]~N(0,1) **S_0[L] ~Uniform categorical**

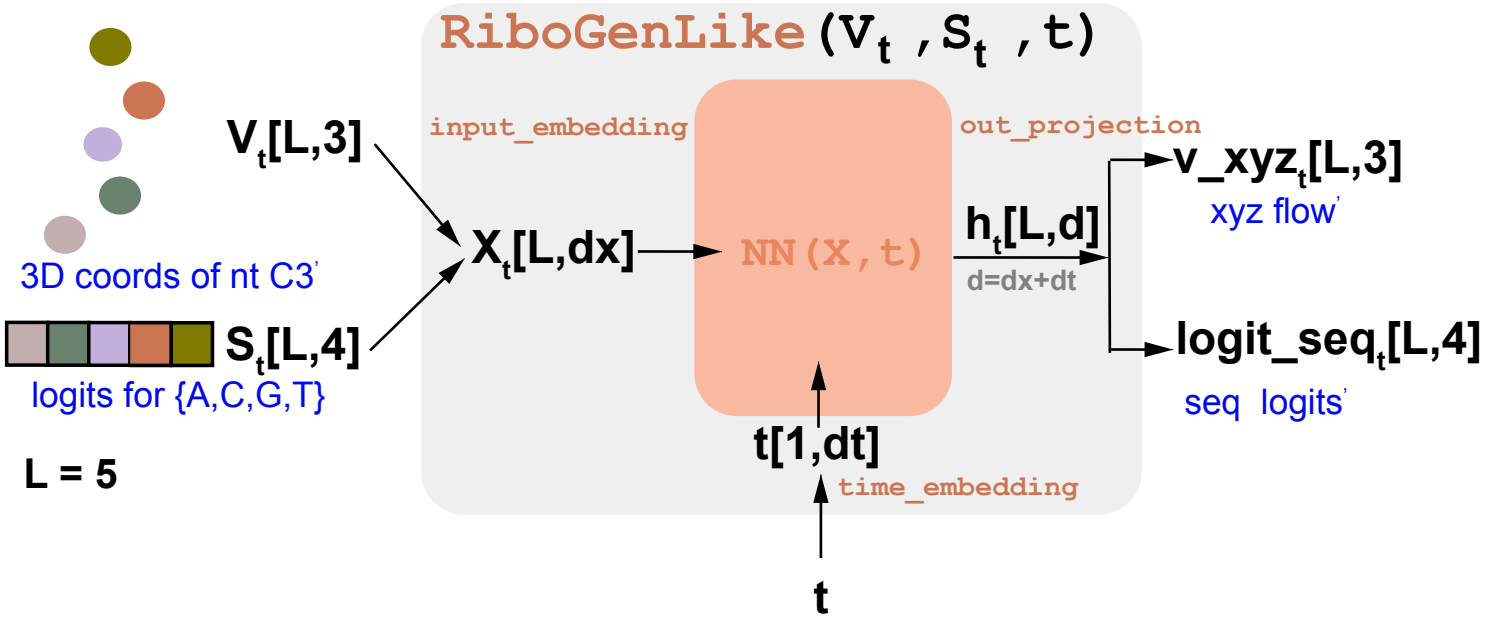


SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   - 2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   - 2.2 Using Euler method:
     $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T} V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
     $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$

V_0[L,3]~N(0,1) **S_0[L] ~Uniform categorical**

`t = k/T`



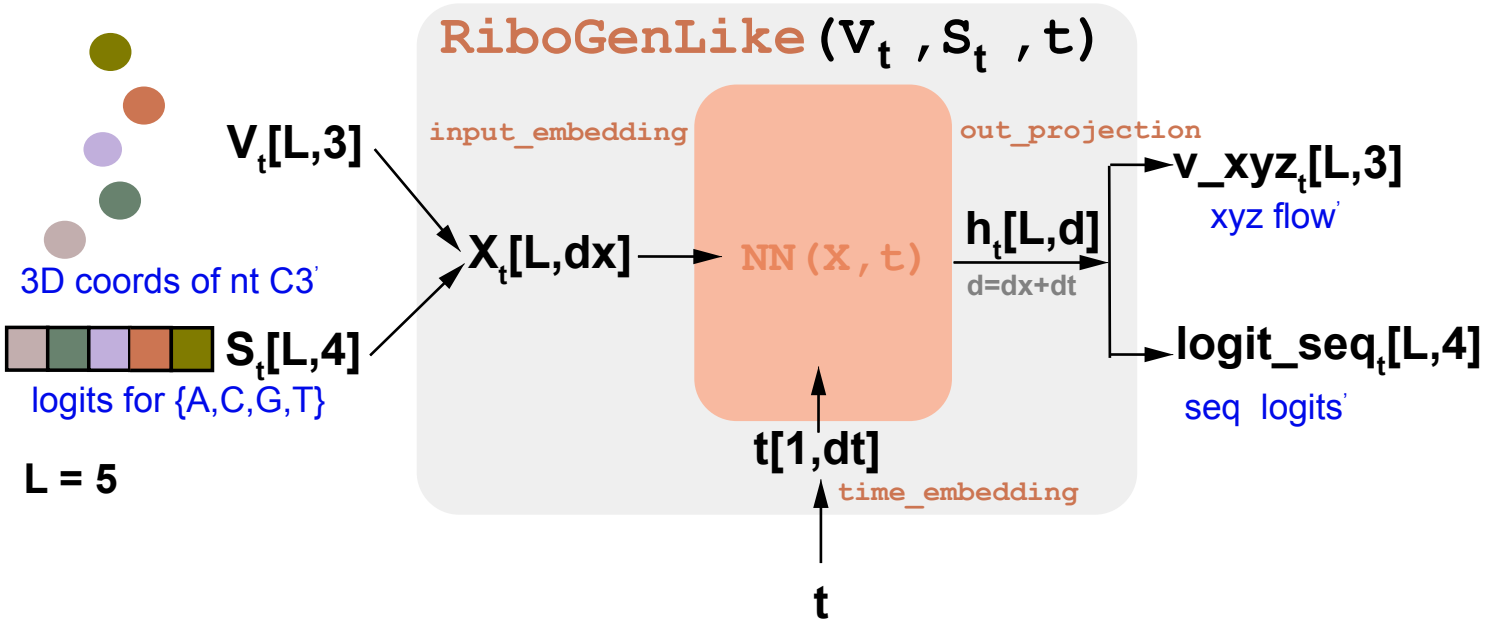
SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   - 2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   - 2.2 Using Euler method:
     $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T} V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
     $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$

V_0[L,3]~N(0,1) **S_0[L] ~Uniform categorical**

t = k/T

= RiboGenLike (V_t , S_t , t)



SAMPLING

1. sample noise $X_0 \sim N(0, I)$
2. In T steps: $k = 1, ..., T$
   - 2.1 Backward Predict flows with NN $V_\theta(X_t)$ for $t = k/T$
   - 2.2 Using Euler method:
     $X_k^{pred} = X_{k-1}^{pred} + \frac{1}{T} V_\theta(X_{k-1}^{pred}) \quad 1 \le k \le T$
     $X_1 = X_0 + \int_0^1 V_\theta(X_t)dt = X_{k=T}^{pred}$

$V_0[L,3] \sim N(0,1)$ $S_0[L]$ ~Uniform categorical

`t = k/T`

= `RiboGenLike` (V_t , S_t , t)

$V_{t+1}[L,3] = V_t[L,3] + 1/T\ v\_xyz_t[L,3]$

$S_{t+1}[L,4] = S_t[L,4] + 1/T\ logit\_seq_t[L,4]$