Generative Models for Inverse Imaging Problems

From mathematical foundations to physics-driven applications

CONTENTS

- Introduction
- Foundations for inverse imaging problems with generative models
  - VAEs
  - GANs
  - Score-based generative models
- Physics-driven applications
  - Generative modeling for cryo-EM analysis
  - Score-based generative models for sparse-view CT and accelerated MRI
- Summary and future outlook

Introduction

In computational imaging, an image sensor measurement $\boldsymbol{y} \in \mathcal{Y}$ from an underlying unknown image $\boldsymbol{x} \in \mathcal{X}$ is usually described by

$$\boldsymbol{y} = \boldsymbol{H}(\boldsymbol{x}) + \varepsilon \tag{1}$$

where ε is measurement noise and $\boldsymbol{H} : \mathcal{X} \mapsto \mathcal{Y}$ is a forward mapping arising from the imaging physics.

- Penalized least squares

computationally expensive iterative solvers

highly dependent upon the regularization

- Deep learning (DL)

usually based on supervised training

- Generative neural network

potential to be used for unsupervised learning

learn the distribution of the real-world data

can incorporate the relevant physical laws and constraints of the measurement process

Foundations for inverse imaging problems with generative models

- $P(\mathrm{x}|\mathrm{y}) = \frac{P(y|x)P(x)}{P(y)}$ Bayesian approach
- The measurement likelihood $P(y|x)$ is sometimes available from the physics of the measurement process.
- The prior distribution $P(x)$ is usually hard to model for computational imaging problems.
- To model the unknown prior distributions, we need to use data-driven methods. **Generative models** are especially useful in learning the prior distribution.

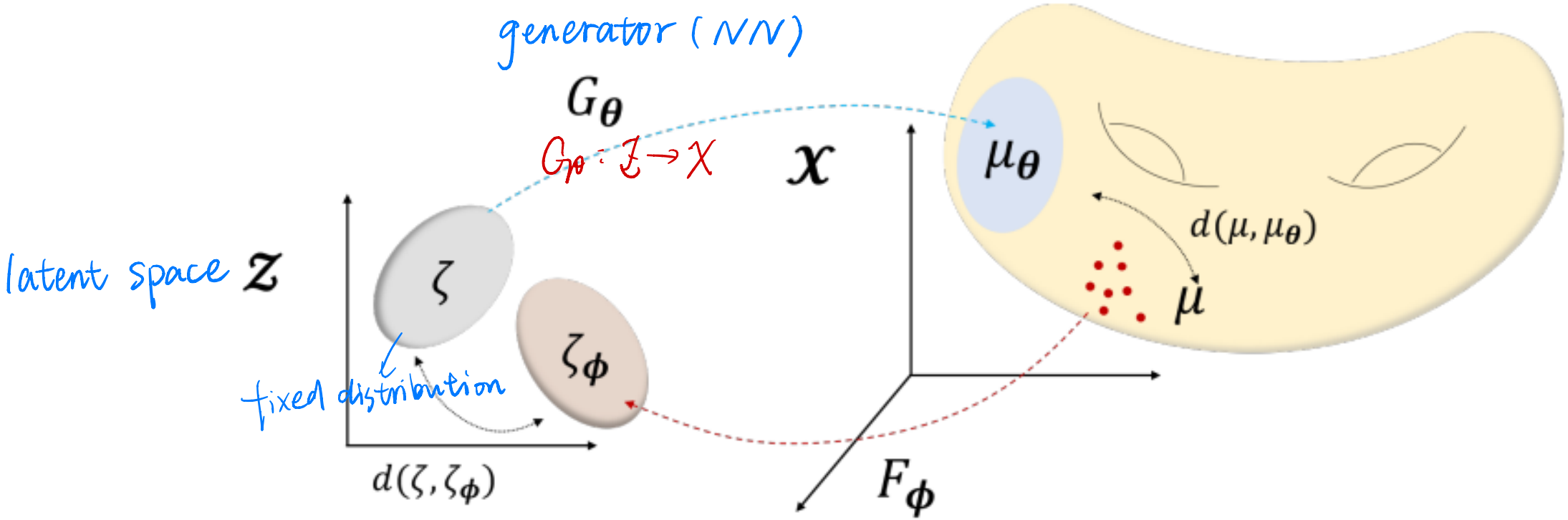

Fig. 5. Geometric view of deep generative models. Fixed distribution ζ in $\mathcal{Z}$ is pushed to μ_θ in $\mathcal{X}$ by the network G_θ, so that the mapped distribution μ_θ approaches the real distribution μ. In VAE, G_θ works as a decoder to generate samples, while F_ϕ acts as an encoder, additionally constraining ζ_ϕ to be as close to ζ. With such geometric view, auto-encoding generative models (e.g. VAE), and GAN-based generative models can be seen as variants of this single illustration.

VAEs (Variational AutoEncoder)

Encoder: $x \to z$, is used only in learning θ from data

Decoder: $z \to x$, $z \sim p(z)$ prior $\xrightarrow{\text{Decoder}}$ $p_\theta(x|z)$ $\xrightarrow{(3)}$ $p_\theta(x)$

The standard VAE [6] models learn the data distribution $p_\theta(\boldsymbol{x})$ by assuming that the data are generated by some random process involving an unobserved continuous random variable $\boldsymbol{z}$ in the latent space $\mathcal{Z}$, i.e.,

$$p_\theta(\boldsymbol{x}) = \int_{\mathcal{Z}} p_\theta(\boldsymbol{x},\boldsymbol{z})\,d\boldsymbol{z} = \int_{\mathcal{Z}} p_\theta(\boldsymbol{x}\,|\,\boldsymbol{z})\underline{p(\boldsymbol{z})}\,d\boldsymbol{z}. \tag{3}$$

prior

To match the distribution $p_\theta(\boldsymbol{x})$ of generated samples to the true unknown data distribution $p(\boldsymbol{x})$, we wish to learn (identify) the model parameters $\boldsymbol{\theta}$ from training data, which can be done, in principle, by maximum likelihood. However, it is hard to learn $\boldsymbol{\theta}$ by directly maximizing the likelihood in (3) since it is intractable to evaluate the integral in (3) in high-dimensional space. Instead of directly evaluating the likelihood, we can compute a lower bound for the log-likelihood of a data point, called the *evidence lower bound* (*ELBO*), or in the context of variational inference, the variational lower bound. This is done using an approximate posterior distribution $q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})$ for the latent variable. As we will see shortly, this is where the probabilistic encoder comes in; its role is to model $q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})$.

$$\begin{aligned}
\log p_\theta(\boldsymbol{x}) &= \log\left(\int p_\theta(\boldsymbol{x}\,|\,\boldsymbol{z})p(\boldsymbol{z})\frac{q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})}{q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})}d\boldsymbol{z}\right) \\
&\overset{(a)}{\geq} \int \log\left(p_\theta(\boldsymbol{x}\,|\,\boldsymbol{z})\frac{p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})}\right)q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})\,d\boldsymbol{z} \\
&= -D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})\,\|\,p(\boldsymbol{z})\right) + \mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}\,|\,\boldsymbol{z})] \\
&\equiv L_{\mathrm{ELBO}}(\boldsymbol{\phi},\boldsymbol{\theta};\boldsymbol{x})
\end{aligned} \tag{4}$$

Jensen

$\log p_\theta(x) = \int_z q_\phi(z|x) \log p_\theta(x)\,dz$

$$L_{\mathrm{ELBO}}(\boldsymbol{\phi},\boldsymbol{\theta};\boldsymbol{x}) = \log p_\theta(\boldsymbol{x}) - D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})\,\|\,p_\theta(\boldsymbol{z}\,|\,\boldsymbol{x})\right). \tag{5}$$

It follows that maximizing the ELBO simultaneously attempts to maximize the data log-likelihood with respect to $\boldsymbol{\theta}$ and minimize the divergence between the true posterior $p_\theta(\boldsymbol{z}\,|\,\boldsymbol{x})$ and its approximant $q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})$.

The approximate posterior $q_\phi(z|x)$ is assumed to be a multivariate normal distribution with diagonal covariance, and the latent variable z is reparameterized as

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon \tag{6}$$

where $\varepsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and $\odot$ denotes the Hadamard product or elementwise product. As shown in Figure 1, the encoder (realized by an NN parameterized by ϕ) outputs the mean $\boldsymbol{\mu}_\phi(x)$ and the standard deviation $\boldsymbol{\sigma}_\phi(x)$ given the input x.

Assuming that the prior $p(z)$ is a standard normal distribution, the loss function for training the VAE model end to end is to minimize the negative ELBO with respect to $\boldsymbol{\theta}, \phi$

$$\begin{aligned}\mathcal{L}_{\mathrm{VAE}}(\boldsymbol{\theta}, \phi) &= \mathbb{E}_{x\sim p(x)}[-L_{\mathrm{ELBO}}(\phi, \boldsymbol{\theta}; x)] \\ &\approx \sum_{i=1}^{N}\Bigg[\frac{1}{2}\left(\left\|\boldsymbol{\mu}_\phi(x^{(i)})\right\|_2^2 + \left\|\boldsymbol{\sigma}_\phi(x^{(i)})\right\|_2^2 - 2\left\|\log(\boldsymbol{\sigma}_\phi(x^{(i)}))\right\|_1\right. \\ &\qquad \left. -1\right) - \frac{1}{L}\sum_{l=1}^{L}\log p_{\boldsymbol{\theta}}(x^{(i)}|z^{(i,l)})\Bigg]\end{aligned} \tag{7}$$

- Advantage

easy and fast to train

- Limitation

$q_\phi(z|x)$ is often restricted to simple distributions

inherent discrepancy between $q_\phi(z|x)$ and $p_{\boldsymbol{\theta}}(z|x)$ remains

it cannot provide the exact likelihood

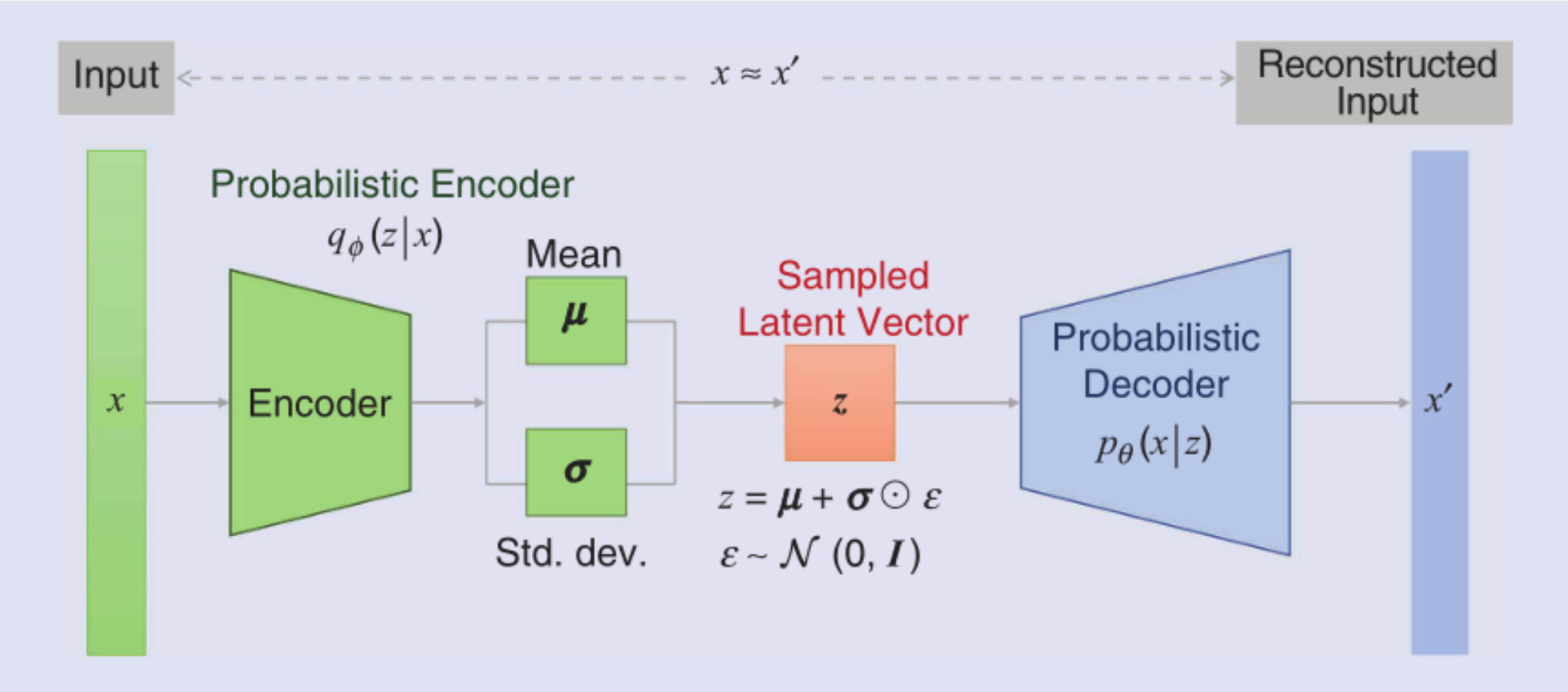

FIGURE 1. The standard VAE structure with probabilistic encoder and decoder. The sampled latent vector is constructed through the reparameterization trick in (6). Std. dev.: standard deviation.

GANs (Generative Adversarial Network)

A GAN [7], [8], [23] (Figure 2) is composed of a generator $G_{\boldsymbol{\theta}}$, which, when driven by Gaussian random vectors $\boldsymbol{z} \sim N(\mathbf{0}, \boldsymbol{I})$, generates samples from the generator distribution p_{gen}, and a discriminator $D_{\boldsymbol{\phi}}$ that compares the data distribution p_{data} with p_{gen}. Both are implemented by NNs, with parameters $\boldsymbol{\theta}, \boldsymbol{\phi}$. GANs pose generative modeling as a problem of minimizing a statistical distance or divergence between probability distributions. The f-GAN [23] uses the f-divergence as the measure of the divergence between two distributions P and Q with density function p and q

$$D_f(P \| Q) = \int_{\mathcal{X}} q(\boldsymbol{x}) f\left(\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right) d\boldsymbol{x} \tag{8}$$

where the function $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex lower-semicontinuous function satisfying $f(1) = 0$. For example, taking $f(s) = s \log s$, the f-divergence becomes the KL divergence.

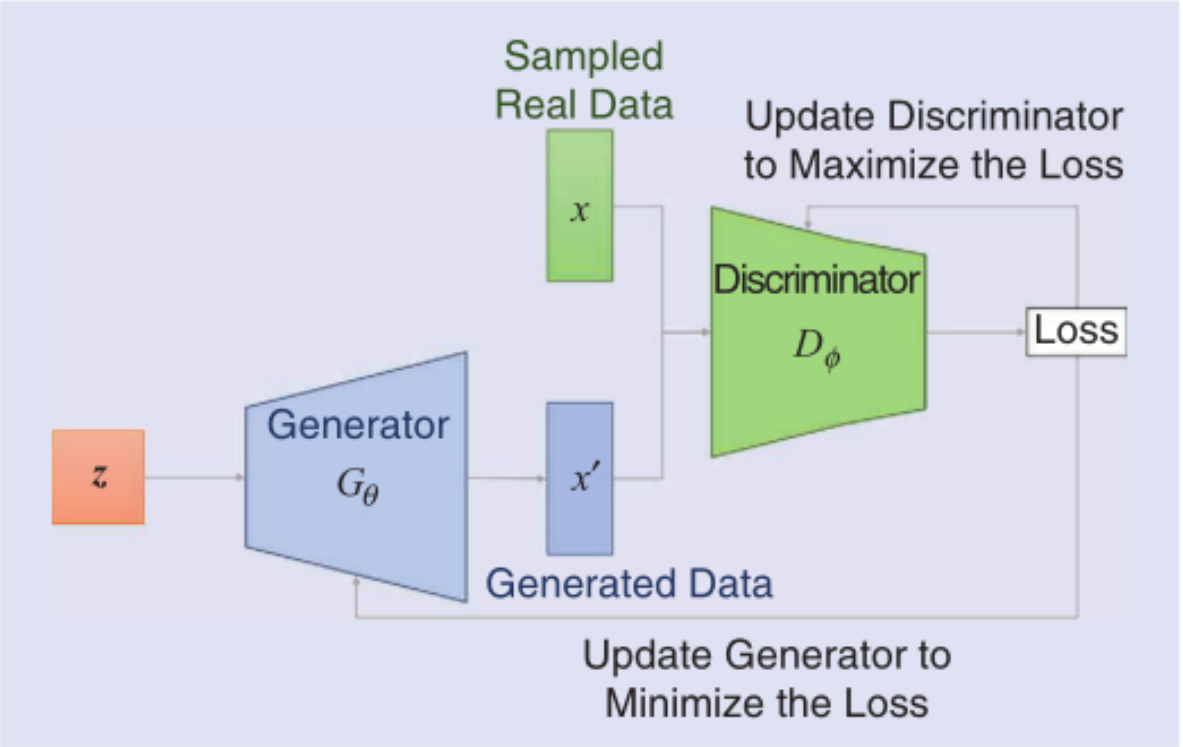

FIGURE 2. The standard GAN architecture.

For D_ϕ, $\max_{D_\phi} \mathcal{L}_{GAN}$ { $D_\phi(x) \to 1$ real ; $D_\phi(G_\theta(z)) \to 0$ fake }

For G_θ, $\min_{G_\theta} \mathcal{L}_{GAN}$, $D_\phi(G_\theta(z)) \to 1$ real

$$\min_{G_\theta} \max_{D_\phi} \mathcal{L}_{\text{GAN}}(D_\phi, G_\theta) \tag{9}$$

where

$$\mathcal{L}_{\text{GAN}}(D_\phi, G_\theta) := \mathbb{E}_{x \sim p_{\text{data}}}[\log D_\phi(\boldsymbol{x})] + \mathbb{E}_z[\log(1 - D_\phi(G_\theta(\boldsymbol{z}))]. \tag{10}$$

The discriminator output $D_\phi(\boldsymbol{x}) \in [0, 1]$, with a value close to one, indicating a sample likely drawn from the data distribution, whereas a value close to zero indicates a "fake" (unlikely) sample. The min-max problem (9) is therefore interpreted as a game between two adversaries—the discriminator, trying to improve its detection of fake samples, and the generator, trying to improve its ability to fool the discriminator.

It was found that GANs using statistical discrepancy measures such as the Jensen–Shannon divergence are especially hard to train, in part because the divergence measure between two distributions "saturates" (to infinity) when their support does not overlap. This scenario is common in the case of image data, which typically lie on (or close to) a low-dimensional manifold in the ambient space, which with high probability does not intersect with the corresponding manifold of the untrained generator. This leads to the vanishing of the gradients of the training loss function, preventing convergence.

- Wasserstein-GAN

For D_ϕ, $\max_{\|D_\phi\|_{Lip}\le 1} L$: $D_\phi(x) \to 1$ real; $D_\phi(G_\theta(z)) \to 0$ fake

For G_θ, $\min_{G_\theta} L$, $D_\phi(G_\theta(z)) \to 1$ real

p-Wasserstein distance between distributions p_{data} and p_{gen} is defined as

$$W_p(p_{\text{data}}, p_{\text{gen}}) = \inf_{\gamma \in \Pi(p_{\text{data}}, p_{\text{gen}})} (\mathbb{E}_{(x,y)\sim\gamma}[\| x - y \|^p])^{1/p} \quad (11)$$

Estimating the Wasserstein distance in high-dimensional space is, however, not straightforward. Arjovsky et al. [8] applied the Kantorovich–Rubinstein duality to the 1-Wasserstein distance, which states

$$W_1(p_{\text{data}}, p_{\text{gen}}) = \sup_{\|f\|_{\text{Lip}} \le 1} \mathbb{E}_{x\sim p_{\text{data}}}[f(x)] - \mathbb{E}_{x\sim p_{\text{gen}}}[f(x)] \quad (12)$$

where the supremum is taken over all 1-Lipschitz functions, $\|f\|_{\text{Lip}} \le 1$, which means that $|f(x_1) - f(x_2)| \le \|x_1 - x_2\|_1$ for all $x_1, x_2 \in \mathcal{X}$. The function f is approximated by a deep NN D_ϕ with parameters ϕ. Therefore, the Wasserstein GAN is formulated as

$$\min_{G_\theta} \max_{|D_\phi|_{\text{Lip}} \le 1} \{\mathbb{E}_{x\sim p_{\text{data}}}[D_\phi(x)] - \mathbb{E}_z[D_\phi(G_\theta(z))]\} \quad (13)$$

and various ways have been suggested to enforce the Lipschitz constraint [24].

However, they're hard to train because of the min-max nature of optimization objectives.

Score-based generative models

Score-based generative models [11], [12], [14] generate new data from noise through learning the gradient of the log probability of the data, also known as the *score function* [29]. To see why score matching is useful in learning a data distribution, let's first consider a probability density function defined in terms of a parameterized function f_θ as $p_\theta(\boldsymbol{x}) = (e^{-f_\theta(\boldsymbol{x})}/Z_\theta)$, where Z_θ (a so-called partition function) is a normalizing constant such that $\int_{\mathcal{X}} p_\theta(\boldsymbol{x})\,d\boldsymbol{x} = 1$. The parameterized function $f_\theta(\boldsymbol{x})$ is often called an *energy-based model*.

$$\boldsymbol{s}_\theta(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) = -\nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x}) - \underbrace{\nabla_{\boldsymbol{x}} \log Z_\theta}_{=0} = -\nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x}). \quad (14)$$

Therefore, the score-based models are trained by minimizing the Fisher divergence between the model and the data distributions, $\mathbb{E}_{p_\theta}\| \boldsymbol{s}_\theta(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x})\|_2^2$.

↓ model ↓ data distribution

approximate the score of $p_\theta(x)$

Another important ingredient in the score-based generative models is modeling the process that transforms the data to noise and its reverse process. diffusion model

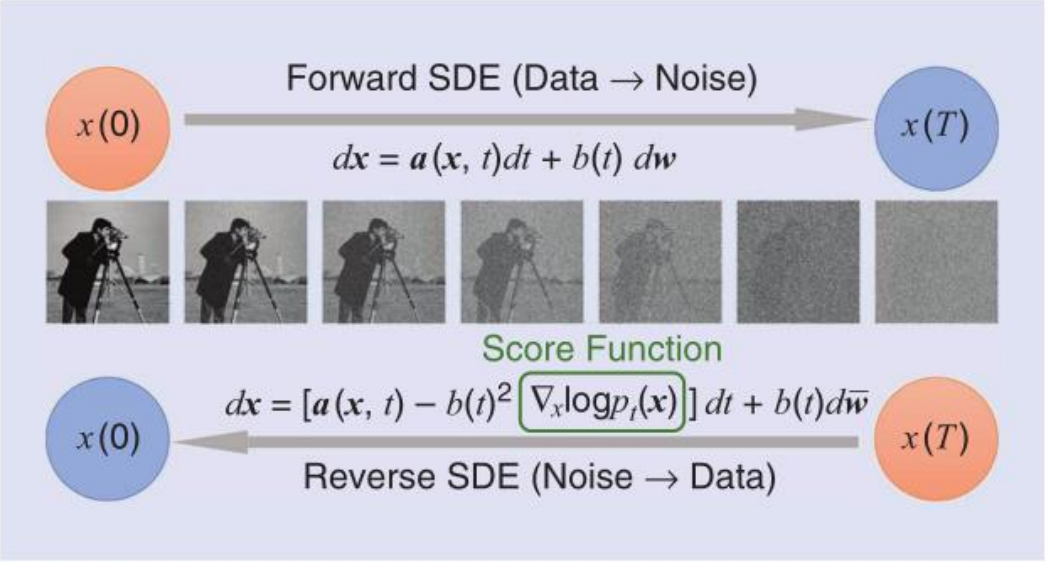

FIGURE 3. An illustration of the score-based generative model. The forward continuous time SDE transforms data to an image sampled from a simple noise distribution. This process can be reversed if we know $\nabla_x \log p_t(\boldsymbol{x})$ (the score function of the distribution) at each intermediate time step. We can sample a data point through evolving a noise image through the reverse-time SDE.

$$P_{0t}(x(t)|x(0)) = \mathcal{N}(x(t)|\alpha(t)x(0), \beta^2(t)I)$$

add noise ↑ image ↑

$$\boldsymbol{x}(t) = \alpha(t)\boldsymbol{x}(0) + \beta(t)\varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{16}$$

The forward process in (16) is used to generate the training data $\boldsymbol{x}(t)$ for $t \in (0, T]$ to learn the gradient of the log probability of the data (score function).

The model is trained by minimizing the weighted Fisher divergence between the model and data distributions and over the time interval $[0, T]$

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim\text{Uniform}(0,T)}[\lambda(t)\mathbb{E}_{\boldsymbol{x}(0)\sim p_{\text{data}}}\mathbb{E}_{\boldsymbol{x}(t)|\boldsymbol{x}(0)} \| s_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t) - \nabla_{\boldsymbol{x}(t)} \log p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0)) \|^2] \tag{18}$$

where the weight $\lambda(t) > 0$ is typically chosen as $1/\lambda(t) \propto \mathbb{E}_{\boldsymbol{x}(0)\sim p_{\text{data}}}\mathbb{E}_{\boldsymbol{x}(t)|\boldsymbol{x}(0)} \| \nabla_{\boldsymbol{x}} \log p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0)) \|_2^2$. In the training, the expectations in the loss function (18) are replaced by empirical means by sampling $\boldsymbol{x}(0)$ from the training data, and for each $\boldsymbol{x}(0)$, sampling $\boldsymbol{x}(t)$ according to (16).

After training, there are a few different ways to sample the distribution and generate new data $\boldsymbol{x}(0) \sim p_{\text{data}}$ from $\boldsymbol{x}(T) \sim p_T$.

Reverse SDE

One is to use a numerical solver for the SDE in (17), such as the Euler–Maruyama approach where dt is approximated by $\Delta t = (T/N)$, with discrete time steps $0 = t_0 < t_1 \cdots < t_N = T$. Another way is to use the predictor–corrector (PC) method. PC sampling uses one step of a numerical solver, which is called the *predictor step*, to generate sample $\boldsymbol{x}(t_{i-1})$ from $\boldsymbol{x}(t_i)$.

A fundamentally different approach to transform noise to data is based on the theoretical result that the reverse-time SDE can be converted into an ordinary differential equation (ODE) without changing its time-evolving distribution $\{p_t(\boldsymbol{x})\}_{t\in[0,T]}$

Physics-driven applications

(Cryo-electron microscopy)

- Generative modeling for cryo-EM analysis

Cryo-EM aims to recover the 3D structure of a particle of interest $\boldsymbol{V}: \mathbb{R}^3 \to \mathbb{R}$ (called *volume*) from a collection of noisy and blurred line-integral projection images $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N$ of the volume taken at unknown projection angles with unknown shifts. These images are obtained by examining a frozen sample containing multiple randomly oriented and shifted copies of the volume. The generation of image $\boldsymbol{y}$ can be modeled as

$$\boldsymbol{y} = \boldsymbol{C}_d * \boldsymbol{S}_t \boldsymbol{P}_R(\boldsymbol{V}) + \boldsymbol{\epsilon} \tag{21}$$

where the $\boldsymbol{P}_R(\boldsymbol{V})$ is the tomographic projection of $\boldsymbol{V}$ rotated by a rotation matrix $\boldsymbol{R} \in SO(3)$, the group of 3D rotations

$$\boldsymbol{P}_R(\boldsymbol{V})(r_x, r_y) = \int_{\mathbb{R}} \boldsymbol{V}(R^\top \boldsymbol{r}) dr_z, \quad \boldsymbol{r} = (r_x, r_y, r_z)^T. \tag{22}$$

St shift the centered 2D projection image by $t=(t_x, t_y)$

The Fourier transform of the resulting image $\hat{\boldsymbol{y}}$ is modulated pointwise by the contrast transfer function (CTF) $\hat{\boldsymbol{C}}_d$, which is determined by the defocus value and other parameters of the electron microscope $\boldsymbol{d}$ and is assumed to be known. In the spatial domain, this effect corresponds to the convolution of the inverse Fourier transform of the CTF with the projection image. We assume that $\boldsymbol{\epsilon}$ is additive WGN with noise variance σ^2. The image size is $D \times D$. According to the Fourier slice theorem, the 2D Fourier transform of the image is given by

$$\hat{\boldsymbol{y}}(k_x, k_y) = \hat{\boldsymbol{C}}_d(k_x, k_y) \tilde{\boldsymbol{S}}_t(k_x, k_y) \hat{\boldsymbol{V}}(\boldsymbol{R}^T [k_x, k_y, 0]^T) + \hat{\boldsymbol{\epsilon}}(k_x, k_y) \tag{23}$$

2D Fourier transform of a projection of a 3D object can be computed as the set of Fourier transforms of the object along lines perpendicular to the direction of the projection.

Single conformation model

The traditional cryo-EM reconstruction [43] uses the maximum a posteriori (MAP) estimate of $\boldsymbol{V}$ from N projection images, marginalizing over the posterior distribution of the $\boldsymbol{\varphi} = (\boldsymbol{R}, t)$

$$\boldsymbol{V}_{\text{rec}} = \underset{\boldsymbol{V}}{\text{argmax}} \sum_{i=1}^{N} \log\left(\int_{SO(3)\times\mathbb{R}^2} p(\boldsymbol{y}_i | \boldsymbol{\varphi}, \boldsymbol{V}) p(\boldsymbol{\varphi}) d\boldsymbol{\varphi} \right) + \log p(\boldsymbol{V}). \tag{24}$$

Multiple conformation model

Structural heterogeneity

In multiclass refinement [43], the image formation model is extended to assume that images are generated from K independent volumes, $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_K$, and the inference requires marginalization over both the pose parameters $\boldsymbol{\varphi}_i$ and the class assignment.

$$\underset{\boldsymbol{V}_1,\ldots,\boldsymbol{V}_K}{\text{argmax}} \sum_{i=1}^{N} \log \sum_{j=1}^{K} \left(\pi_j \int_{SO(3)\times\mathbb{R}^2} p(\boldsymbol{y}_i | \boldsymbol{\varphi}, \boldsymbol{V}_j) p(\boldsymbol{\varphi}) d\boldsymbol{\varphi} \right) + \sum_{j=1}^{K} \log p(\boldsymbol{V}_j). \tag{25}$$

In CryoSPARC [44], the class assignment probabilities are assumed to be uniform, i.e., $\pi_j = 1/K$. This approach requires prior knowledge of the number of classes and is computationally feasible only for a small number of classes. However, the protein conformation changes are continuous and may be poorly approximated with a small number of discrete volumes.

VAE-based generative models

CryoDRGN (Deep Reconstructing Generative Network) [47] employs VAE to learn a continuous low-dimensional manifold over a protein's conformational states from 2D cryo-EM images in an unsupervised way and perform ab initio (i.e., not as a refinement of an existing volume) reconstruction of the volumes.

The model contains a standard VAE probabilistic encoder $q_\phi(z|\hat{y}_i)$.

The Fourier transform $\hat{V}$ of the volume in CryoDRGN is modeled by a probabilistic decoder $p_\theta(\hat{V}|z,k)$ with two separate inputs: 1) the latent variable z from the encoder and 2) the frequency k.

3D coordinate $k \in [-0.5, 0.5]^3$, CryoDRGN use positional encoding $\rho : [-0.5, 0.5]^3 \to [-1, 1]^{3D}$ to map the 3D coordinate to a higher dimensional vector. For $k = (k_1, k_2, k_3)^T$, the positional encoding consists of sine and cosine waves, and the mapping is defined as

$$\rho^{(2i)}(k_j) = \sin\left(k_j D\pi\left(\frac{2}{D}\right)^{\frac{2i}{D}}\right),$$

$$\rho^{(2i+1)}(k_j) = \cos\left(k_j D\pi\left(\frac{2}{D}\right)^{\frac{2i}{D}}\right),$$

$$i = 1, \ldots, \frac{D}{2}; \quad j = 1, 2, 3. \tag{26}$$

It is empirically observed that using this encoding works well for clean data. For noisy data, it is required to exclude 10% of the high-frequency components. The decoder outputs the volume at frequency k as $\hat{V}_z(k) = G_\theta(z, \rho(k))$.

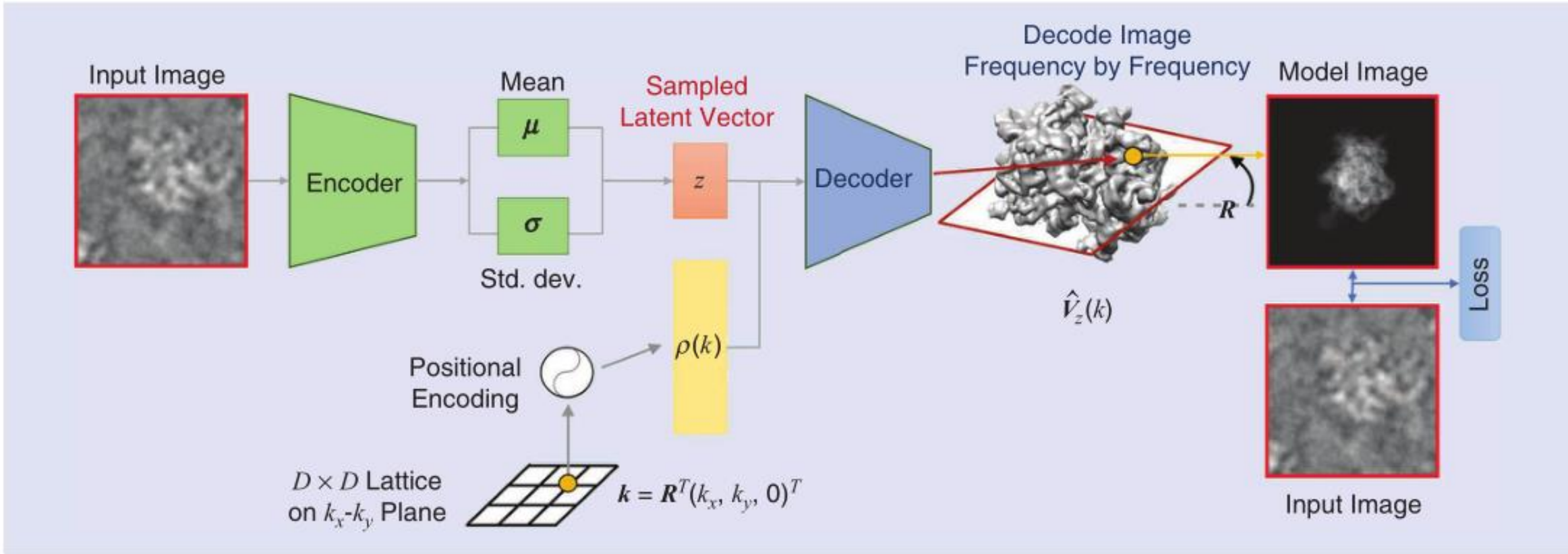

FIGURE 4. The CryoDRGN model architecture. The VAE is used to perform approximate inference for latent variable z denoting structural heterogeneity. The decoder reconstructs an image frequency by frequency in k-space, given z and $\boldsymbol{\rho}(\boldsymbol{k})$, the positional encoding of 3D Cartesian k-space coordinates. The 3D frequency coordinates corresponding to each input image Fourier coefficient are obtained by rotating a $D \times D$ lattice on the $k_x - k_y$ plane by $\boldsymbol{R}$, the orientation of the particle. The latent orientation $\boldsymbol{R}$ for each image is inferred by a branch and bound global optimization procedure. (Source: Figure courtesy of the authors of [37].)

Using the Fourier slice theorem in (23) and the decoder $G_{\boldsymbol{\theta}}$, we can evaluate the negative log-likelihood of the image $\hat{\boldsymbol{y}}$ given the latent variable $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\hat{\boldsymbol{y}})$ and the pose parameters $(\boldsymbol{R}, \boldsymbol{t})$

$$
\begin{aligned}
&-\log p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\boldsymbol{z}, \boldsymbol{R}, \boldsymbol{t}) \\
&= \frac{1}{2\sigma^2} \sum_{k_x} \sum_{k_y} |\hat{\boldsymbol{y}}(k_x, k_y) - \hat{\boldsymbol{C}}_d(k_x, k_y) \tilde{\boldsymbol{S}}_t(k_x, k_y) \\
&\quad \times G_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{\rho}(\boldsymbol{R}^T [k_x, k_y, 0]^T))|^2 + D^2 \log(\sqrt{2\pi}\,\sigma). \qquad (27)
\end{aligned}
$$

CryoDRGN jointly estimates the pose parameters and the VAE network parameters by alternating between updating those two sets of parameters. When the pose parameters $(\boldsymbol{R}, \boldsymbol{t})$ are fixed, CryoDRGN minimizes the negative ELBO using the Adam optimizer [48] with respect to $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ following the standard VAE framework

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \sum_{m=1}^{B} & D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\hat{\boldsymbol{y}}_{n_m}) \| p(\boldsymbol{z})) \\
& - \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\hat{\boldsymbol{y}}_{n_m})} [\log p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}_{n_m}|\boldsymbol{z}, \boldsymbol{R}, \boldsymbol{t})] \qquad (28)
\end{aligned}
$$

where B is the batch size, and $\{n_m\}_{m=1}^{B}$ is a set of random indices for projection images chosen at each training iteration.

The physics-based measurement model is incorporated in the negative log-likelihood in (27), which appears in the negative ELBO (28) for training the VAE model. With fixed VAE network parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, a global search over $SO(3) \times \mathbb{R}^2$ is performed for the maximum-likelihood estimation of the pose $(\boldsymbol{R}, \boldsymbol{t})$ for each image given the decoded volume $\hat{\boldsymbol{V}}_z(\boldsymbol{k}) = G_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{\rho}(\boldsymbol{k}))$. An efficient joint maximum-likelihood estimator pose estimation and a VAE training scheme for cryo-EM are detailed in [49].

After training, CryoDRGN provides per-particle image latent encoding. The encoder network outputs $\boldsymbol{z}_i = \boldsymbol{\mu}_{\boldsymbol{\phi}^*}(\hat{\boldsymbol{y}}_i)$, which corresponds to the approximate MAP estimate of the latent variable $\boldsymbol{z}_i$ given $\hat{\boldsymbol{y}}_i$, i.e., $\boldsymbol{z}_i = \text{argmax}_{\boldsymbol{z}} q_{\boldsymbol{\phi}^*}(\boldsymbol{z}|\hat{\boldsymbol{y}}_i) = \boldsymbol{\mu}_{\boldsymbol{\phi}^*}(\hat{\boldsymbol{y}}_i)$. The trained decoder network can then generate 3D volumes given arbitrary values of the latent variable $\boldsymbol{z}$ via $\hat{\boldsymbol{V}}_z(\boldsymbol{k}) = G_{\boldsymbol{\theta}^*}(\boldsymbol{z}, \boldsymbol{\rho}(\boldsymbol{k}))$.

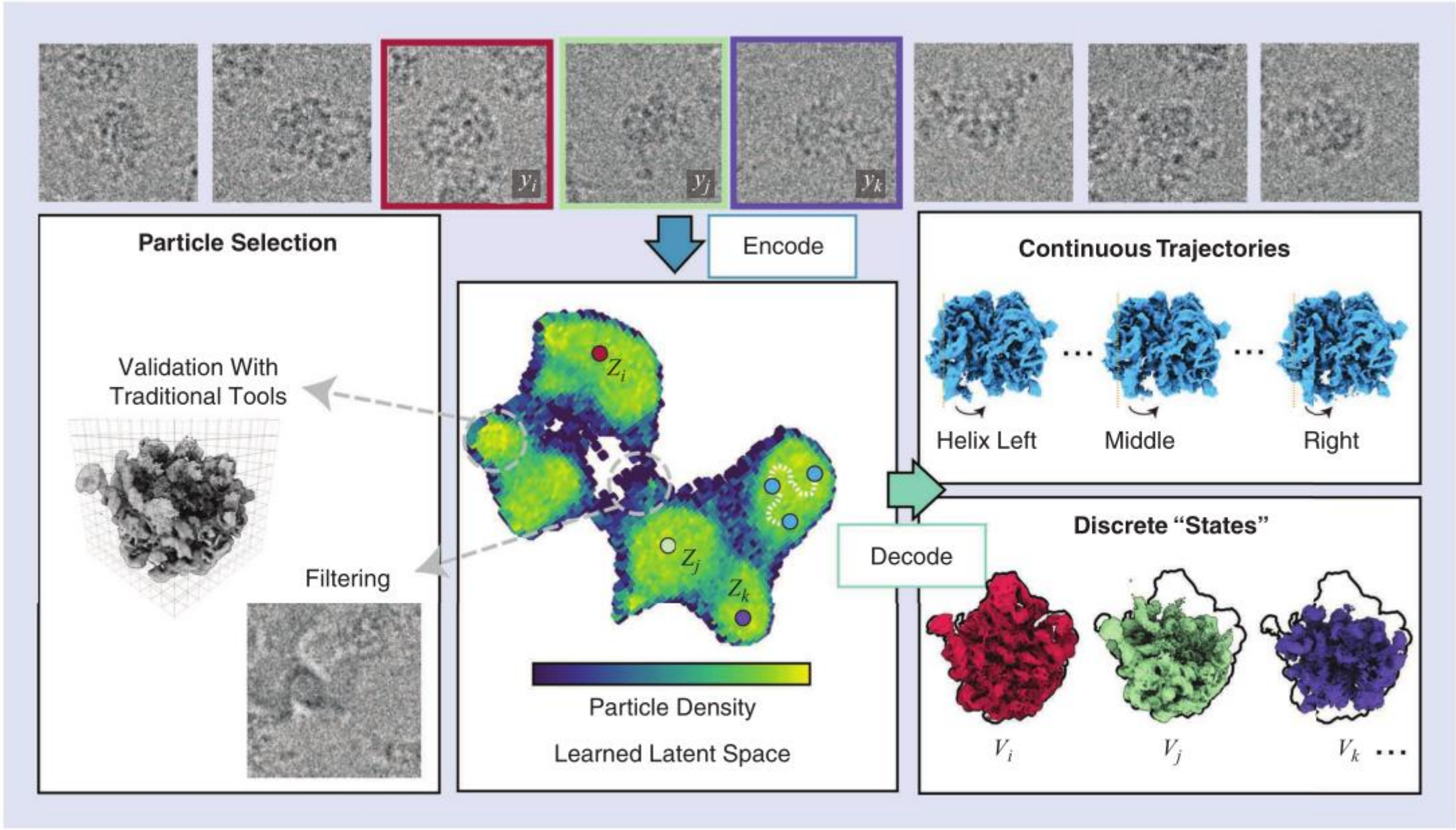

FIGURE 5. A structural heterogeneity analysis. With the trained CryoDRGN encoder, all particle images of the ribosome dataset [51] are encoded into the latent space, as shown in the center panel. The trained decoder can map the latent variable to the 3D volumes, as indicated in the right panel. In addition, the latent space representation can be used to filter particle images and remove impurities. (Source: Figure courtesy of the authors of [47].)

GAN-based generative models

CryoGAN [4] and Multi-CryoGAN [18] address the 3D reconstruction problem in cryo-EM from the perspective of distribution matching. These methods adopt the Wasserstein GAN structure and use a physics-based cryo-EM data simulator to generate the projection images. Unlike the maximum likelihood-based approaches, the adversarial learning-based approaches do not require the estimation of the individual pose parameters. Instead, in CryoGAN [4] and Multi-CryoGAN [18], the imaging parameters (rotations, in-plane translations, and CTF parameters) $\boldsymbol{\varphi} = (\boldsymbol{R}, \boldsymbol{t}, \boldsymbol{d})$ are drawn from a known prior distribution $p_{\boldsymbol{\varphi}}$.

CryoGAN is used to reconstruct a single 3D volume from the image data, assuming a single conformation. Given the 3D volume $\boldsymbol{V}$, the distribution of the generated images is denoted by $p_{\text{gen}}(\boldsymbol{y}; \boldsymbol{V})$. The reconstruction is achieved by minimizing the 1-Wasserstein distance between $p_{\text{gen}}(\boldsymbol{y}; \boldsymbol{V})$ and $p_{\text{data}}(\boldsymbol{y})$ and becomes the following min-max optimization problem:

Single conformation

$$V_{\text{rec}} = \underset{\boldsymbol{V}}{\operatorname{argmin}} \max_{\|D_\phi\|_{\text{Lip}} \le 1} \left(\mathbb{E}_{\boldsymbol{y} \sim p_{\text{data}}}[D_\phi(\boldsymbol{y})] - \mathbb{E}_{\boldsymbol{y} \sim p_{\text{gen}}(\boldsymbol{y}; \boldsymbol{V})}[D_\phi(\boldsymbol{y})] \right). \tag{29}$$

The Lipschitz constraint $\|D_\phi\|_{\text{Lip}}$ is enforced by penalizing the norm of the gradient of D_ϕ with respect to its input. Based on empirical samples, the loss in (29) is reformulated as

$$\mathcal{L}(\boldsymbol{V}, \boldsymbol{\phi}) = \sum_{m=1}^{B} \left(D_{\boldsymbol{\phi}}(\boldsymbol{y}_{\text{data}}^{n_m}) - D_{\boldsymbol{\phi}}(\boldsymbol{y}_{\text{gen}}^{m}) - \lambda \left[\left(\|\nabla_{\boldsymbol{y}} D_{\boldsymbol{\phi}}(\boldsymbol{y}_{\text{int}}^{m})\| - 1 \right)^2 \right] \right) \tag{30}$$

where B is the batch size; $\{n_m\}_{m=1}^{B}$ is a set of random indices for projection images selected at each iteration; $\boldsymbol{y}_{\text{gen}}^{m}$ denotes the projections from the current estimate $\boldsymbol{V}$ generated by the cryo-EM physics simulator according to (21); and $\lambda \in \mathbb{R}_+$ is a gradient penalty coefficient. The interpolated sample used in the gradient penalty is $\boldsymbol{y}_{\text{int}}^{m} = a_m \boldsymbol{y}_{\text{data}}^{n_m} + (1 - a_m) \boldsymbol{y}_{\text{gen}}^{m}$, where a_m is sampled from a uniform distribution on the interval $[0, 1]$. Using the interpolated samples ensures that the learned discriminator function is 1-Lipschitz on the domain spanned by both the generated and real data.

To address the multiconformation setting, Multi-CryoGAN [18] (Figure 6) adds a convolutional NN (CNN) G_γ to the CryoGAN architecture to learn a mapping from a latent space to the 3D conformation distribution. It samples a latent vector z from a prior distribution p_z. Then, a CNN G_γ maps the input latent variable z to the conformation manifold, i.e., $\boldsymbol{V}_z = G_{\boldsymbol{\gamma}}(z)$. Based on the generated volume $\boldsymbol{V}_z$, the cryo-EM physics-based simulator generates noisy projection images. The distribution of the generated projection images is denoted by $p_{\text{gen}}(\boldsymbol{y}; G_{\boldsymbol{\gamma}})$. To find the network parameters in the generator G_γ, Multi-CryoGAN minimizes the 1-Wasserstein distance between $p_{\text{data}}(\boldsymbol{y})$ and $p_{\text{gen}}(\boldsymbol{y}; G_{\boldsymbol{\gamma}})$, which results in the following min-max optimization problem.

$$\begin{aligned}\gamma^* &= \operatorname*{argmin}_{\boldsymbol{\gamma}} W_1(p_{\text{data}}, p_{\text{gen}}) \\ &= \operatorname*{argmin}_{\boldsymbol{\gamma}} \max_{\|D_\phi\|_{\text{Lip}} \le 1} \left(\mathbb{E}_{\boldsymbol{y}\sim p_{\text{data}}}[D_{\boldsymbol{\phi}}(\boldsymbol{y})] - \mathbb{E}_{\boldsymbol{y}\sim p_{\text{gen}}(\boldsymbol{y};G_\gamma)}[D_{\boldsymbol{\phi}}(\boldsymbol{y})]\right).\end{aligned} \tag{31}$$

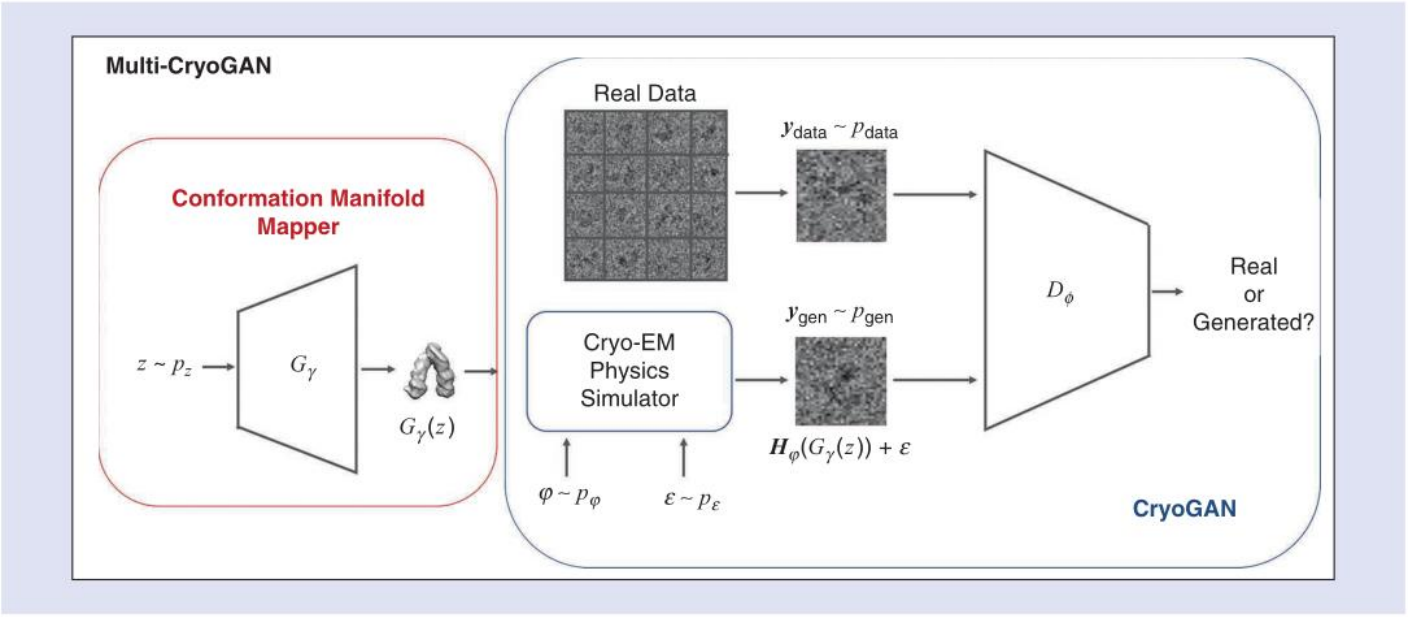

FIGURE 6. The model architecture for Multi-CryoGAN, which contains the architecture for (a) conformation manifold mapper (in a red box) and (b) CryoGAN (in a blue box). (Source: Figure courtesy of the authors of [18].)

- Score-based generative models for sparse-view CT and accelerated MRI

In sparse-view CT and accelerated MRI, we consider the general linear measurement model

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\varepsilon} = \mathcal{P}(\boldsymbol{\Lambda})\boldsymbol{M}\boldsymbol{x} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{42}$$

where $\boldsymbol{M}$ corresponds to the Radon transform in sparse-view CT and Fourier transform in accelerated (undersampled) MRI, respectively, and $\boldsymbol{\Lambda}$ is an $n \times n$ diagonal matrix where $\boldsymbol{\Lambda}_{ii} \in \{0, 1\}$ with $\mathrm{tr}(\boldsymbol{\Lambda}) = m$, and $\mathcal{P}(\boldsymbol{\Lambda}) \in \{0, 1\}^{m \times n}$ is an operator that keeps only the rows of $\boldsymbol{\Lambda}$ that $\boldsymbol{\Lambda}_{i,i} \neq 0$. Matrix $\boldsymbol{A}$ is assumed to have full row rank.

In the "Score-Based Generative Models" section, we discussed the unconditional sampling of $\boldsymbol{x}(t)$ to generate data $\boldsymbol{x}(0)$ from noise $\boldsymbol{x}(T)$ by following the reverse-time SDE. Given the measurement data $\boldsymbol{y}$, in principle, the reconstruction should be achieved by generating approximate samples from the conditional stochastic process $\{\boldsymbol{x}(t) \mid \boldsymbol{y}\}$ from $t = T$ to $t = 0$. However, it is difficult to directly solve the conditional reverse-time SDE $\{\boldsymbol{x}(t) \mid \boldsymbol{y}\}_{t \in [0,T]}$ without using paired training data to learn the conditional score function. For simplicity, we set $T = 1$ for the discussions later.

To overcome the difficulty in sampling directly from $\{\boldsymbol{x}(t) \mid \boldsymbol{y}\}$, [1] introduced the following stochastic process $\{\boldsymbol{y}(t)\}_{t\in[0,1]}$, given the unconditional stochastic process $\{\boldsymbol{x}(t)\}_{t\in[0,1]}$

$$\begin{aligned}\boldsymbol{y}(t) &\triangleq \boldsymbol{A}\boldsymbol{x}(t) + \alpha(t)\boldsymbol{\varepsilon} \overset{(a)}{=} \boldsymbol{A}(\alpha(t)\boldsymbol{x}(0) + \beta(t)\boldsymbol{z}) + \alpha(t)\boldsymbol{\varepsilon} \\ &\overset{(b)}{=} \alpha(t)\boldsymbol{y} + \beta(t)\boldsymbol{A}\boldsymbol{z} \end{aligned} \tag{43}$$

where $\boldsymbol{z} \in \mathbb{R}^n \sim \mathcal{N}(0, \boldsymbol{I})$. Equality (a) follows from the Gaussian transition kernel $p_{0t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(0)) = \mathcal{N}(\boldsymbol{x}(t) \mid \alpha(t)\boldsymbol{x}(0), \beta^2(t)\boldsymbol{I})$

Therefore, it is tractable to generate samples $\tilde{\boldsymbol{y}}(t) \sim p_t(\boldsymbol{y}(t) \mid \boldsymbol{y})$ according to (43). In addition, we can modify the iterative sampling algorithm designed for the unconditional stochastic process $\{\boldsymbol{x}(t)\}_{t\in[0,1]}$ by encouraging data consistency between the samples $\tilde{\boldsymbol{x}}(t) \sim p_t(\boldsymbol{x}(t))$ and the samples $\tilde{\boldsymbol{y}}(t)$ at each time t by constructing the intermediate samples

$$\begin{aligned}\tilde{\boldsymbol{x}}'(t) = \underset{\boldsymbol{v}\in\mathbb{R}^n}{\operatorname{argmin}} \{(1-\lambda)\| \boldsymbol{v} - \tilde{\boldsymbol{x}}(t)\|_{\boldsymbol{M}}^2 + \lambda \min_{\boldsymbol{u}\in\mathbb{R}^n} \| \boldsymbol{v} - \boldsymbol{u}\|_{\boldsymbol{M}}^2\}, \\ \text{subject to } \boldsymbol{A}\boldsymbol{u} = \tilde{\boldsymbol{y}}(t)\end{aligned} \tag{44}$$

where $\lambda \in [0, 1]$ is a parameter that balances between data consistency with the unconditional generation $\tilde{\boldsymbol{x}}(t)$ and the conditional generation $\tilde{\boldsymbol{y}}(t)$. The weighted norm $\|\cdot\|_{\boldsymbol{M}}^2$ is defined as $\|\boldsymbol{x}\|_{\boldsymbol{M}}^2 = \|\boldsymbol{M}\boldsymbol{x}\|_2^2$.

When $\lambda = 0$, we have $\tilde{\boldsymbol{x}}'(t) = \boldsymbol{x}(t)$, and we don't incorporate the stochastic process of $\boldsymbol{x}(t)$ associated with the observation $\boldsymbol{y}$. This parameter can be tuned on a validation dataset using Bayesian optimization. In summary, the physics of the measurement process is incorporated into the conditional generation through two steps: 1) define the process $\{\boldsymbol{y}(t)\}$ according to (43) and 2) encourage data consistency according to (44).

The sampling process selects a sequence of times $0 = t_0 < t_1 < \cdots < t_N = 1$ and iterates according to

$$\tilde{\boldsymbol{y}}(t_i) = \alpha(t_i)\boldsymbol{y} + \beta(t_i)\boldsymbol{A}\boldsymbol{z}_i, \quad \boldsymbol{z}_i \in \mathbb{R}^n \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{45a}$$

$$\begin{aligned}\tilde{\boldsymbol{x}}'(t_i) = \boldsymbol{M}^{-1}[&\lambda\boldsymbol{\Lambda}\mathcal{P}^\dagger(\boldsymbol{\Lambda})\tilde{\boldsymbol{y}}(t_i) + (1-\lambda)\boldsymbol{\Lambda}\boldsymbol{M}\tilde{\boldsymbol{x}}(t_i) \\ &+ (\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{M}\tilde{\boldsymbol{x}}(t_i)]\end{aligned} \tag{45b}$$

$$\tilde{\boldsymbol{x}}(t_{i-1}) = \tilde{\boldsymbol{x}}'(t_i) - \frac{a(t_i)\tilde{\boldsymbol{x}}'(t_i)}{N} + \frac{b(t_i)^2 \boldsymbol{s}_{\boldsymbol{\theta}^*}(\tilde{\boldsymbol{x}}'(t_i), t)}{N} + \frac{b(t_i)\boldsymbol{z}_i}{\sqrt{N}} \tag{45c}$$

where $\mathcal{P}^\dagger(\boldsymbol{\Lambda})$ denotes the right inverse of $\mathcal{P}(\boldsymbol{\Lambda})$. The process runs from $t = 1$ backward to $t = 0$ and draws approximate samples $\tilde{\boldsymbol{x}}'(0)$ from $p(\boldsymbol{x} \mid \boldsymbol{y})$ as the reconstructed images. This is especially useful in quantifying the uncertainty of the reconstructions by directly evaluating the mean and variance of the reconstructed images for the same measurement $\boldsymbol{y}$.

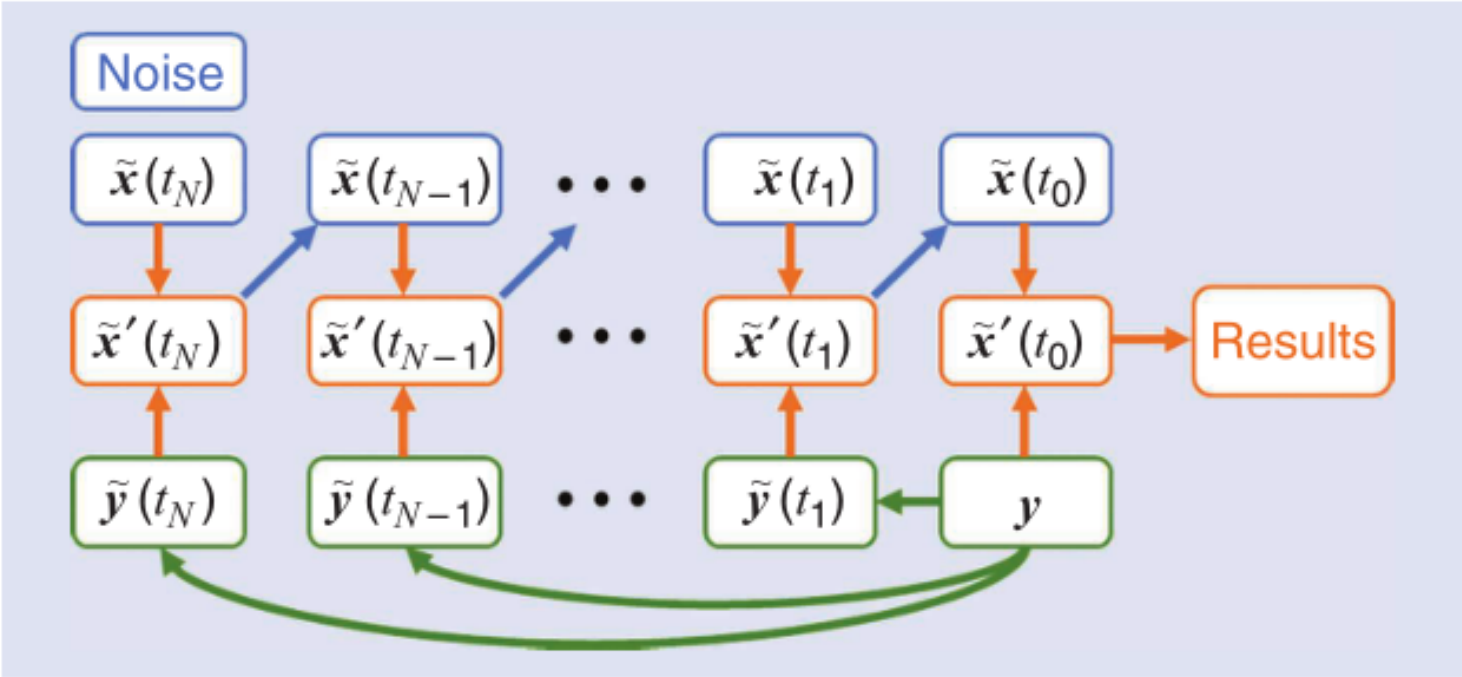

FIGURE 9. An illustration of the iterative sampling method in [1] for solving the inverse problems. The green arrows indicate the process of generating samples $\tilde{\boldsymbol{y}}(t_i)$ from the observation $\boldsymbol{y}$ according to (45a). The orange arrows show the process of combining the unconditional generative samples $\tilde{\boldsymbol{x}}(t_i)$ with $\tilde{\boldsymbol{y}}(t_i)$ according to (45b). The blue arrows indicate the reverse-time stochastic process in (45c).

Summary and future outlook

- cryo-EM
  - more systematic quantitative comparisons to choices and hyperparameters
  - provide physical or biological meaning to the distance between the latent space of conformation variables
  - still lack a standardized measure or gold-standard task to evaluate how well a method is able to capture the heterogeneit
- score-based approaches
  - slow convergence